MASTER EN SCIENCES INFORMATIQUES

La carte à microprocesseur CP8. Description et évaluation. Réalisation d'un logiciel d'aide à la programmation d'applications

De Pra, Maurizio

Award date:
1987

Awarding institution:
Universite de Namur

Facultés Universitaires Notre-Dame de la Paix, Namur
Institut d'Informatique
Année académique 1986 - 1987

LA CARTE A MICROPROCESSEUR CP8

- Description et évaluation

- Réalisation d'un logiciel d'aide à la programmation d'applications

Maurizio DE PRA

Mémoire présenté en vue de l'obtention du grade de
Licencié et Maître en informatique

REMERCIEMENTS

Mes plus vifs remerciements vont à mon promoteur J. Ramaekers et à son assistante C. Stas-Mahiat pour l'attention qu'ils ont portée à ce mémoire tout au long de sa réalisation.

Je tiens également à exprimer ma gratitude à la direction de BULL CP8 et à tout le personnel des services Etudes Logiques et Logiciel qui m'ont permis de réaliser un stage enrichissant en dépit de conditions difficiles.

Enfin, je tiens à remercier D. Basyn et H. Leemans de la société ASCII pour les propositions constructives relatives au logiciel réalisé dans le cadre de ce mémoire.

TABLE DES MATIERES

Chapitre 3 : Description technique

INTRODUCTION

Au cours de ces dernières années, les organismes bancaires et commerciaux ont mis l'accent sur la promotion d'un mode de paiement particulier : le paiement par carte. Dans ce contexte, on parle de plus en plus de la carte à microprocesseur qui, bien qu'utilisée de façon marginale actuellement, semble promise à un bel avenir. De nombreux constructeurs, tant européens que japonais et américains, se sont déjà lancés dans la production et la commercialisation de ce type de cartes.

Le propos de ce mémoire est de présenter une des cartes à microprocesseur développées et commercialisées par le constructeur français BULL CP8. Cette carte est la carte Masque 4. La présentation se divise en deux parties.

La première partie commence par situer le contexte de la carte à microprocesseur. Ensuite viennent une description technique relativement générale de la carte présentée et une évaluation de cette carte par rapport à trois critères qui nous semblent fondamentaux : la protection de l'information, la polyvalence et l'aspect multi-services. Signalons que la description technique ne constitue pas un manuel d'utilisation de la carte mais que son seul but est de poser les idées de base nécessaires à la compréhension de l'évaluation de la carte.

La deuxième partie traite de la réalisation d'un logiciel d'aide à la programmation d'applications basées sur la carte CP8. Cette deuxième partie constitue un dossier d'analyse qui comprend la présentation des objectifs assignés au logiciel, la description de ses fonctionnalités, la découpe du logiciel en modules et, enfin, la spécification détaillée de ces modules.

Signalons pour terminer cette introduction que la première partie a été réalisée avec la collaboration d'Yves Triniane, étudiant à l'Institut d'Informatique, tandis que la seconde est personnelle et a été réalisée sur demande de la société ASCII.

PREMIERE PARTIE

DESCRIPTION ET EVALUATION DE LA CARTE A MICROPROCESSEUR CP8

I. GENERALITES

1.1. Qu'est-ce qu'une carte à microprocesseur?

La carte à microprocesseur fait partie des cartes à microcircuit qui sont elles-mêmes englobées dans la famille des cartes à mémoire. Toute carte à mémoire a pour fonction de mémoriser de l'information et de la protéger de manière à éviter la fraude. Il existe trois types de cartes à mémoire utilisées actuellement :

- la carte embossée,
- la carte à pistes magnétiques,
- la carte à microcircuit.

1.1.1. Carte embossée

Elle est constituée d'une simple carte plastique (PVC) sur laquelle des informations (nom, adresse, numéro de compte du porteur, ...) sont gravées par un procédé d'embossage.

La capacité de mémorisation de cette carte reste très faible .

Le niveau de sécurité offert par cette carte est également très faible. En effet, toute information inscrite sur la carte est directement lisible à l'oeil nu. Il n'est donc pas envisageable d'y graver des informations confidentielles. De plus, il est relativement aisé de fabriquer de fausses cartes.

1.1.2. Carte à pistes magnétiques

Elle est constituée d'une carte plastique (PVC) sur laquelle ont été ajoutées des pistes magnétiques. Les informations peuvent être stockées sur les pistes magnétiques ou sur la carte (par embossage comme précédemment).

La capacité de mémorisation reste relativement faible.

Le niveau de sécurité augmente considérablement par rapport à la carte embossée. En effet, l'accès aux informations mémorisées par les pistes magnétiques ne peut être réalisé qu'en disposant d'un lecteur-encodeur. De

plus les informations peuvent être chiffrées, ce qui garantit un certain degré de confidentialité.

1.1.3. Carte à microcircuit

Elle est constituée d'une carte plastique (PVC) dans laquelle a été inséré un composant électronique (puce). Les informations peuvent être mémorisées dans la puce ou sur le PVC. Les informations stockées dans la puce peuvent être chiffrées ou non.

La capacité de mémorisation de ce type de cartes évolue constamment. Actuellement, elle est de l'ordre de quelques Koctets.

Il existe trois types de cartes à microcircuit :

- -la carte à mémoire simple,
- -la carte à logique câblée,
- -la carte à microprocesseur.

1.1.3.1. Carte à mémoire simple

La puce est ici un simple composant mémoire qui ne permet pas de réaliser d'autres fonctions que la lecture ou l'écriture d'informations.

Le niveau de sécurité offert par la carte à mémoire simple est analogue à celui offert par la carte à pistes magnétiques : ici aussi, il suffit de disposer d'un lecteur-encodeur pour avoir accès à l'information stockée dans la puce.

1.1.3.2. Carte à logique câblée

La puce est constituée de mémoire et de circuits câblés qui s'interposent entre la mémoire et les lecteurs-encodeurs.

Le niveau de sécurité augmente considérablement puisqu'il n'est plus possible d'accéder directement à la mémoire : il faut, au préalable, demander une autorisation d'accès aux circuits câblés. Ces derniers, possédant une intelligence câblée, sont en mesure d'accorder ou non l'autorisation en fonction du contexte.

1.1.3.3. Carte à microprocesseur

La puce est constituée d'un microprocesseur, d'une mémoire ROM, et d'une mémoire de stockage. Le microprocesseur et la mémoire ROM ont la même fonction que des circuits câblés, c'est-à-dire protéger les accès à la mémoire de stockage. Les cartes à microprocesseur et à logique câblée sont donc semblables au niveau du fonctionnement et au niveau de la sécurité offerte. La différence est que l'intelligence de la carte à microprocesseur n'est pas câblée mais contenue sous forme de programmes dans la mémoire ROM (ce point sera largement détaillé au chapitre 3).

Remarque : pour toutes les cartes à microcircuit, la mémoire de stockage doit être non volatile. On ne peut donc pas utiliser de mémoire RAM traditionnelle. Nous verrons par la suite qu'une mémoire de type PROM convient très bien.

1.2. Pourquoi une carte à microprocesseur ?

Ce sont principalement des impératifs sécuritaires et économiques qui sont à la base de la carte à microprocesseur.

1.2.1. Impératifs de sécurité

Parallèlement à la généralisation de l'utilisation des cartes à mémoire pour le paiement, la fraude sur ces mêmes cartes a connu un taux de croissance particulièrement élevé dans certains pays (surtout aux Etats-Unis). Cette fraude revêt diverses formes :

- vols de cartes, avant ou après leur remise au porteur,
- duplication de cartes à partir d'une carte volée,
- fabrication de fausses cartes,
- .....

1.2.2. Impératifs économiques

- L'utilisation des modes de paiement traditionnels (chèques, virements, ...) entraîne des charges considérables pour les organismes bancaires et leurs clients car l'enregistrement des paiements implique généralement des traitements manuels. L'utilisation de la carte à microprocesseur permet l'automatisation des traitements et donc une réduction des coûts.

- Il est toujours difficile de garantir une bonne sécurité pour des transactions informatisées s'effectuant loin d'un centre de traitement : il est souvent indispensable d'avoir recours à des réseaux reliant tous les points de service au centre de traitement le plus proche. La carte à microprocesseur, grâce à son intelligence active, prend en charge la gestion de nombreuses procédures de sécurité, ce qui rend souvent facultative l'utilisation de réseaux. Cette solution entraîne de fortes réductions dans le coût des communications dans les pays géographiquement étendus.

1.3. Typologie des services

Si la carte à microprocesseur a d'abord été conçue pour résoudre des problèmes de paiement, on s'est très vite aperçu qu'elle constituait aussi un support adéquat pour d'autres types de services. Il existe actuellement quatre domaines privilégiés pour l'utilisation de la carte à microprocesseur :

- le paiement électronique,
- le dossier portable,
- le contrôle d'accès,
- la sécurité des systèmes d'information.

1.3.1. Le paiement électronique

Ce domaine recouvre le télépaiement et le prépaiement.

Le télépaiement consiste à enregistrer auprès de l'organisme de paiement le montant d'une transaction réalisée par le porteur de la carte (retrait d'argent, achat de marchandises, ...). L'enregistrement d'une transaction peut se faire automatiquement via un réseau (système on-line) ou passer par un support informatique (bande magnétique, ...) qui sera transmis périodiquement à l'organisme de paiement (système off-line). Les informations concernant la transaction sont également enregistrées sur la carte pour permettre au porteur de garder une trace des opérations qu'il a effectuées et le cas échéant, de régler d'éventuels différends avec son organisme de paiement.

Le prépaiement sert à supprimer la monnaie pour les paiements de montants faibles. Deux cas peuvent se présenter : le porteur achète des unités de service qui seront consommées au fur et à mesure des besoins (exemple: la carte de téléphone), ou la carte sert d'abonnement électronique c'est-à-dire que la carte est utilisée comme un abonnement classique dont la validité sera contrôlée automatiquement dans chaque point de service (exemple : abonnement autoroute en France).

1.3.2. Le dossier portable

Dans ce cas, la carte sert de support pour le stockage d'informations relatives au porteur. Ces informations sont enregistrées soit uniquement sur la carte, soit sur la carte et dans un fichier central. Dans les deux cas, le dossier portable garantit la disponibilité des informations sans avoir recours

à des réseaux. L'apport de la carte dans ce domaine est donc principalement d'ordre économique.

1.3.3 Le contrôle d'accès

La carte est utilisée ici comme matérialisation de droit d'accès à des sites (laboratoires, bureaux, ...) ou à des systèmes d'information. Si on considère l'accès à des sites, la carte fait office de clé et le lecteur-encodeur fait office de serrure.

Un premier apport se situe au niveau de la sécurité. Comme nous l'avons déjà signalé, la carte prend en charge la gestion de nombreuses procédures de sécurité qui sont traditionnellement gérées par des médias tels que terminaux, réseaux, ... Dans ce domaine, la carte offre une plus grande résistance à la fraude que la plupart des médias généralement utilisés.

Le second apport se situe au niveau de la convivialité. La carte, grâce à sa capacité de mémorisation, permet de réaliser des connexions automatiques. Par exemple, l'accès à un service via Minitel (France), nécessite un dialogue entre l'utilisateur et le terminal. Pour un service donné, le dialogue est toujours le même. La carte permet de mémoriser les réponses que l'utilisateur doit fournir au terminal et de les communiquer automatiquement au terminal à chaque demande de service.

1.3.4. La sécurité des systèmes d'information

Dans le domaine relativement vaste de la sécurité des systèmes d'information, la carte offre des perspectives intéressantes au niveau du contrôle d'accès (cfr supra) et des problèmes de transmission.

Pour assurer une bonne sécurité aux transmissions, il faut d'une part, éviter que des informations soient interceptées et d'autre part, éviter que des informations soient modifiées accidentellement ou par malveillance.

On utilise dans le premier cas la cryptographie (procédé mathématique pour chiffrer un message afin qu'il soit inintelligible pour ceux à qui il n'est pas destiné) et dans le second, la signature électronique (procédé qui permet de garantir qu'un message n'a pas été altéré et qu'il émane d'une source bien définie).

Nous verrons plus tard que la carte permet de réaliser des systèmes de chiffrement et de signature électronique particulièrement fiables.

Remarque : cette classification des types de services n'est pas rigoureuse et n'est donnée qu'à titre indicatif. En pratique, une service peut chevaucher plusieurs des catégories ci-dessus. Par exemple, l'accès à une base de données via un réseau de télécommunications peut mettre en oeuvre simultanément des procédures de sécurité de systèmes d'information et de télépaiement.

II. CRITERES D'EVALUATION D'UNE CARTE A MICROPROCESSEUR

Les caractéristiques des cartes à microprocesseur peuvent varier d'un constructeur à l'autre. Toutefois, il est souhaitable que la réalisation d'une carte, quel qu'en soit le constructeur, soit guidée par trois objectifs :

- la protection de l'information,
- la polyvalence,
- l'aspect multi-services.

Dans ce chapitre, nous allons préciser ces trois notions que nous considérons comme les principaux critères d'évaluation d'une carte à microprocesseur. Nous allons également présenter rapidement les grands principes adoptés par BULL CP8 au niveau de la protection de l'information.

2.1. Protection de l'information

La protection de l'information constitue l'objectif principal d'une carte à miroprocesseur. Elle consiste à lutter contre les altérations volontaires ou involontaires de l'information. Il est d'usage de relier le mot sécurité aux actions malveillantes et le mot sûreté aux actions accidentelles.

La sécurité vise à garantir trois qualités de l'information :

- la confidentialite c'est-à-dire le secret,
- l'authenticité c'est-à-dire l'identification des sources (signature),
- l'intégrité c'est-à-dire l'absence de modifications dues à une tentative de fraude.

La sûreté vise à garantir deux qualités de l'information :

- la pérennité c'est-à-dire l'absence de modifications dues à une cause accidentelle (erreur de transmission, ...),
- l'exactitude c'est-à-dire l'absence d'erreurs dans la représentation du réel perçu.

Le domaine d'action de la carte à microprocesseur est principalement la sécurité, mais il ne faut pas oublier que sécurité et sûreté sont souvent intimement liées.

Les mesures de sécurité implémentées grâce à la carte CP8 se situent tant au niveau physique qu'au niveau logique.

Remarque : la carte à microprocesseur ne peut garantir à elle seule la sécurité d'une application. Les concepteurs d'application doivent veiller à implémenter certains dispositifs de sécurité complémentaires (gestion de listes noires de cartes volées, ...). Dans la suite de ce mémoire, nous ne parlerons que des procédures de sécurité entièrement (partiellement) gérées par la carte ou par les appareils constituant l'environnement technique de la carte (lecteur-encodeur, ...).

2.1.1. Sécurité physique

Les dispositifs de sécurité implémentés au niveau physique visent à interdire tout accès à la mémoire de stockage non autorisé par le microprocesseur. En d'autres termes, il doit être impossible de lire, écrire ou effacer des informations sans avoir obtenu au préalable l'autorisation du microprocesseur. La sécurité physique est donc directement liée à la technologie de fabrication de la puce et à son architecture interne.

2.1.2. Sécurité logique

La sécurité logique recouvre tous les aspects de la sécurité gérés par les programmes de la carte et par les programmes des appareils dialogant avec la carte. Ces programmes implémentent cinq fonctions fondamentales, appelées fonctions de sécurité :

- l'authentification de l'utilisateur,
- l'authentification de la carte,
- la certification,
- la signature électronique,
- la génération de clés de chiffrement.

2.1.2.1. Authentification de l'utilisateur

Le probleme posé est de savoir si la carte est utilisée par une personne autorisée (mesure contre les cartes volées). En fonction des opérations à réaliser, cette personne peut être le porteur (possesseur) de la carte ou son émetteur.

La carte authentifie l'utilisateur en comparant une clé proposée par l'utilisateur avec une clé stockée dans la carte.

2.1.2.2. Authentification de la carte

Le problème consiste, pour un système externe (ordinateur central ou simple lecteur-encodeur de cartes), à s'assurer qu'il dialogue avec une vraie carte et que cette carte peut accéder au service demandé par l'utilisateur.

Le sytème peut s'en assurer en demandant à la carte d'exécuter un calcul faisant intervenir une clé secrète interne à la carte. Seule une vraie carte peut posséder une clé interne correcte et donc fournir un résultat correct au calcul demandé.

2.1.2.3. Certification

La certification consiste à fournir un certificat unique et infraudable relatif à une information déterminée. Ce certificat permet de garantir qu'une information se trouve matériellement à un endroit déterminé de la carte et qu'elle y est correctement inscrite.

Par exemple, à l'occasion d'une transaction monétaire, il faut vérifier que cette opération est bien enregistrée dans la carte, de manière à éviter toute contestation éventuelle à propos de cette transaction.

2.1.2.4. Signature électronique

La signature électronique permet de vérifier l'authenticité, l'intégrité et la pérennité des informations au cours d'une transmission.

Le principe est le suivant :

- L'émetteur condense les informations à transmettre en une chaîne de quelques caractères, à laquelle il joint une référence à l'origine du message (signature). Il génère ensuite un certificat en chiffrant la chaîne de caractères et la référence. Ce certificat est alors joint au texte transmis.

- Au départ du texte reçu et de la référence à l'origine du texte, le récepteur calcule un certificat de la même manière qu'à l'émission. Il compare ensuite le certificat qu'il a calculé et celui qu'il a reçu. Si les

deux certificats sont identiques, on peut affirmer que le message reçu est authentique et qu'il n'a pas été modifié pendant sa transmission.

2.1.2.5. Génération de clés de chiffrement

Le problème le plus délicat en matière de chiffrement est constitué par le transport des clés. Comment être sûr qu'à aucun moment la clé de chiffrement ne pourra être interceptée? La carte permet de résoudre ce problème en supprimant le transport des clés. L'idée est que lémetteur et le récepteur génèrent chacun les clés de chiffrement, ce qui rend leur transport inutile et leur interception impossible.

La génération de clés comporte deux étapes :

- l'émetteur génère un nombre aléatoire qu'il transmet au récepteur.

- l'émetteur et le récepteur chiffrent le nombre aléatoire en utilisant une clé secrète commune. Sous sa forme chiffrée, le nombre aléatoire constitue une clé de chiffrement.

Grâce à l'utilisation de nombres aléatoires, une seule clé secrète commune peut être utilisée pour générer un nombre quelconque de clés de chiffrement.

Remarque : la génération de clés ne résoud pas complètement le problème du transport de clés dans la mesure où, pour chiffrer le nombre aléatoire, il faut disposer au préalable d'une clé de chiffrement commune. Il faut donc veiller à ce que cette clé ne puisse pas être interceptée pendant son transport. Nous verrons ultérieurement les mesures de sécurité qui protègent les clés communes.

2.2. Polyvalence

Comme nous l'avons vu au chapitre 1, la carte peut être utilisée pour divers services (paiement électronique, dossier portable, ...). De plus, au sein d'un même type de service, il peut y avoir des différences considérables. Pour des raisons de coût, il est indispensable que la carte soit assez souple pour pouvoir être utilisée indifféremment pour n'importe quel type de service.

2.3. Aspect multi-services

Comme la capacité de la mémoire des puces ne cesse de croître, il devient possible d'utiliser une seule carte simultanément pour des services différents. La mémoire disponible est alors partagée pour permettre la mise en oeuvre des différents services.

L'intérêt d'une carte multi-services est évident : elle permet d'éviter aux grands consommateurs de services la possession de plusieurs cartes et, par conséquent, la mémorisation de plusieurs clés secrètes.

III. DESCRIPTION TECHNIQUE

3.1. Architecture interne de la puce

Bien que nous l'appellions communément microprocesseur, la puce utilisée par la carte est bien plus qu'un simple microprocesseur classique; il s'agit en fait d'un microsystème informatique comportant :

- un microprocesseur 6805 ou 8048,
- une mémoire PROM (Programmable Read Only Memory),
- une mémoire ROM (Read Only Memory),
- une mémoire RAM (Random Acces Memory),
- une interface d'entrée/sortie.

Cet ensemble microprocesseur-mémoires est monolithique, c'est-à-dire intégré dans un seul et même composant. Il est illustré à la figure 3.1.

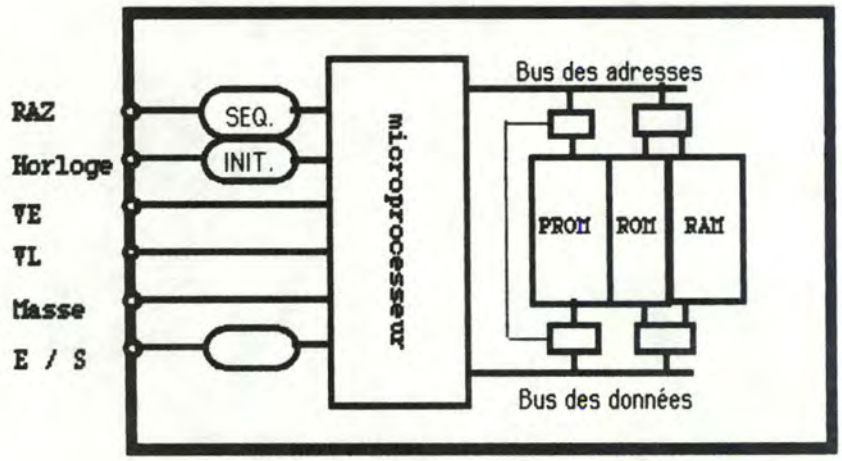


Fig. 3.1 : L'architecture interne de la puce

3.1.1. Le microprocesseur

Ce microprocesseur 8 bits est capable d'exécuter les programmes inscrits dans sa mémoire ROM. Il est autoprogrammable, c'est-à-dire que, contrairement aux architectures classiques, il peut réaliser lui-même l'écriture de sa mémoire PROM à n'importe quel moment.

3.1.2. La mémoire PROM

Elle est utilisée pour le stockage des données propres à chaque application (transaction, dossier portable, ...). Dans la suite de ce mémoire, elle sera indifféremment désignée par mémoire PROM ou par mémoire de stockage.

Il faut noter que cette mémoire possède quelques particularités :

- A la fabrication, l'ensemble des bits de cette mémoire est au niveau logique 1. L'écriture consiste à positionner au niveau logique 0 les bits désirés. Tout bit positionner à 0 ne peut plus revenir à l'état 1.

- Pour écrire dans cette mémoire, il faut que la tension d'écriture soit plus élevée que dans le cas d'une lecture des mémoires (d'où la présence dans l'architecture d'un second fil d'alimentation tolérant des tensions plus élevées).

- Cette mémoire PROM, est en fait une mémoire EPROM dont les possibilités d'effacement ne sont pas utilisées.

- A l'avenir, l'EPROM sera remplacée par une EEPROM. Les mémoires EEPROM permettront au processeur d'effacer sa mémoire de façon sélective c'est-à-dire qu'il pourra n'effacer qu'une partie bien définie de la mémoire (par exemple un mot). Il gérera alors l'opération d'effacement comme une opération d'écriture ou de lecture .

3.1.3. La mémoire ROM

Cette mémoire contient l'intelligence de la carte sous forme de programmes. Cet ensemble de programmes peut varier d'un type de carte à un autre et est appelé MASQUE. Il réalise les fonctionnalités suivantes :

- la gestion de la mémoire de stockage,
- une fonction logico-mathématique,
- la gestion des accès à la mémoire de stockage,
- le jeu d'instructions du microprocesseur,
- la gestion des échanges entre la carte et le monde extérieur.

3.1.4. La mémoire RAM

C'est la mémoire de travail du microprocesseur. Elle est volatile et ne constitue donc pas un moyen de stockage durable. Elle est utilisée uniquement pour les entrées/sorties et le stockage des informations internes au microprocesseur . Elle est totalement inaccessible de l'extérieur.

3.1.5. Interface physique de la puce

L'interface physique est constituée de six fils :

- RAZ : une tension appliquée sur ce fil déclenche l'initialisation physique et logique du composant. Suite à cette remise à zéro, la carte envoie des informations sur ses caractéristiques et son état.

- HORLOGE : le microprocesseur n'ayant pas d'horloge interne, celle-ci est externe. Une base de temps de 3,6 MHZ doit être fournie à la carte par ce fil.

- VL : sur ce fil est appliquée la tension d'alimentation du composant, suffisante pour les opérations de lecture des mémoires RAM, ROM et PROM. Cette tension est d'environ 5 volts. La carte ne peut rester sous cette tension plus de 30 secondes.

- VE : sur ce fil est appliquée, à la demande de la carte, la tension nécessaire à la programmation de la PROM. Cette tension est d'environ 21 volts et sa durée maximale de 5 secondes.

- ZERO : fil de masse.

- E/S : c'est par ce fil que transitent les données échangées entre la carte et le monde extérieur. Le mode d'échange est asynchrone - sous forme caractère - et sa vitesse est de 9600 bauds.

Ces six fils sont reliés par soudure à une pastille qui permet d'établir la connexion entre la puce et les systèmes externes (fig. 3.2 et 3.3).

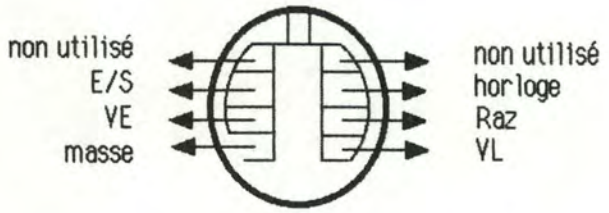


Fig. 3.2 : La pastille

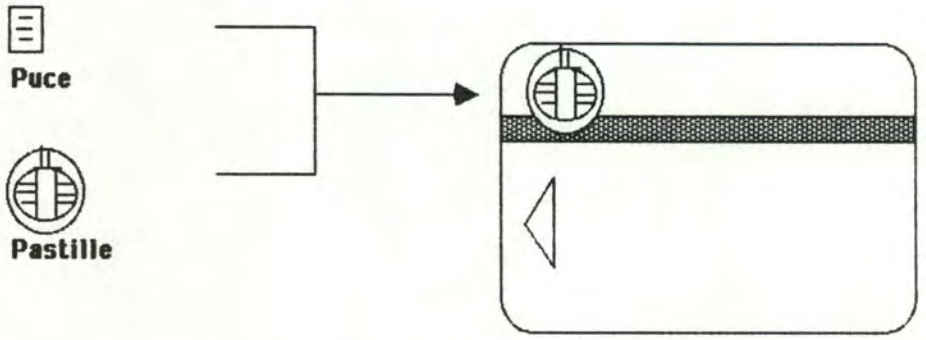


Fig. 3.3 : La carte à microprocesseur

Pour établir la connexion entre la puce et les systèmes externes, la carte doit être insérée dans un connecteur. Celui-ci est spécifiquement conçu pour alimenter électriquement les cartes à microprocesseur et pour permettre le dialogue. Le connecteur est lui-même inséré dans un lecteur-encodeur par lequel transitent toutes les communications. Ces communications seront vues plus en détail au point 3.3. (les programmes d'application).

Remarque : comme nous l'avons déjà signalé, la taille des mémoires ne cesse de croître. Voici un bref historique des types de puce utilisés avec leurs principales caractéristiques.

TYPE DE LA PUCE	MAM01	MAM02	MAM03	MAM04
MICROPROCESSEUR	6805	8048	6805	
MEMOIRE RAM	36 oct.	44 oct.	52 oct.	
MEMOIRE ROM	1600 oct.	2048 oct.	2048 oct.	
MEMOIRE PROM	1024 oct.	1024 oct.	1920 oct.	-> 8192

Fig. 3.4 : Caractéristiques des puces

3.2. Description d'un masque

Rappelons que le masque est constitué de l'ensemble des programmes situés en ROM et qu'il réalise essentiellement :

- la gestion de la mémoire de stockage,
- une fonction logico-mathématique,
- la gestion des accès à la mémoire de stockage,
- le jeu d'instructions du microprocesseur,
- la gestion des échanges entre la carte et le monde extérieur.

Nous allons décrire successivement ces cinq aspects du masque. Cette description n'est valable que pour la phase d'utilisation de la carte par son porteur. En effet, nous verrons au paragraphe 3.4 (Cycle de vie de la carte) que la gestion de la mémoire de stockage et la gestion des accès à cette mémoire sont différentes avant et après la mise en service de la carte.

3.2.1. Gestion de la mémoire de stockage PROM

La mémoire de stockage est logiquement divisée en 7 zones distinctes qui se subdivisent elles-mêmes en mots.

3.2.1.1. Les zones

Le découpage de la mémoire de stockage PROM est donné à la figure 3.5.

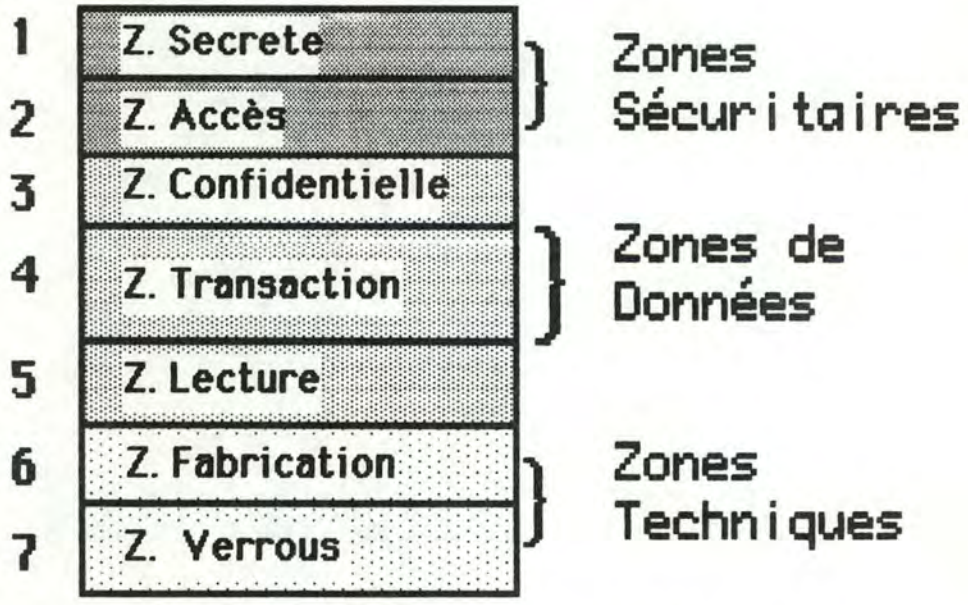


Fig. 3.5 : Les zones de la mémoire PROM

Les zones se répartissent en trois classes :

- les zones sécuritaires contenant des informations utiles au microprocesseur pour gérer l'accès à la mémoire PROM.

- les zones de données contenant les informations propres à l'application et au porteur de la carte .

- les zones techniques contenant des informations nécessaires à la gestion de la mémoire PROM.

Détaillons chacune des zones :

a) La zone secrète

La zone secrète ou zone des clés est une zone hautement protégée. En effet seul le microprocesseur peut en lire le contenu qui est une liste de clés :

- clé émetteur primaire,
- clé emetteur secondaire,
- clé porteur de type I,
- clé porteur de type II,
- clé interne,
- clé de fabrication.

Il est important pour bien comprendre le rôle de ces clés de situer les utilisateurs de la carte :

- L' **émetteur** est un organisme qui propose un service à des clients.

  Il peut exister deux émetteurs pour une même carte. Dans ce cas, les deux émetteurs se partagent la mémoire disponible.

  Chaque émetteur doit pouvoir fixer les modalités d'utilisation de la carte (plafonds périodiques, ...) pour le service qu'il propose et il doit être le seul à pouvoir fixer ces modalités. Dans ce but , chaque émetteur dispose d'une clé qui lui est propre : **clé émetteur primaire, clé émetteur secondaire.**

- Le **porteur** est la personne qui utilise la carte pour un service proposé par un émetteur. Il possède une clé confidentielle qui l'identifie, appelée clé porteur.

  Initialement, c'est l'émetteur qui choisit la clé porteur , mais celle-ci peut être modifiée par le porteur. Il peut donc exister deux clés porteur pour une même carte : **clé porteur type I, type II** (à tout instant, une seule clé porteur est opérationnelle).

Voyons maintenant la fonction de chaque clé :

- clé émetteur (primaire ou secondaire) : elle permet à l'émetteur de fixer les modalités d'utilisation de la carte pour le service qu'il propose. Cette clé permet également l'accès aux informations protégées.

- clé porteur : elle permet au porteur de faire usage des droits fixés par l'émetteur (réaliser un achat chez le commerçant par exemple) et d'avoir accès aux informations protégées.

- clé interne : elle est utilisée comme paramètre de la fonction logico-mathématique de la carte et permet notamment l'authentification de la carte ainsi que la certification.

- clé de fabrication : elle est utilisée pour protéger l'accès à la mémoire PROM avant la mise en service de la carte. L'intérêt de cette clé sera mis en évidence au point 3.4 (Cycle de vie de la carte).

b) La zone d'accès

La zone d'accès est utilisée par le microprocesseur afin de mémoriser toutes les tentatives d'accès (correctes ou non) à une zone protégée c'est-à-dire nécessitant une présentation de clé.

La présentation d'un certain nombre de clés fausses entraîne le blocage de la carte (3 clés porteur fausses, 1 clé émetteur fausse). Lorsqu'une carte est bloquée, certaines opérations deviennent irréalisables (par exemple, l'accès à des informations protégées est impossible).

Une carte peut être débloquée (recyclée) après présentation des clés porteur et émetteur correctes .

La figure 3.6 illustre les états possibles d'une carte .

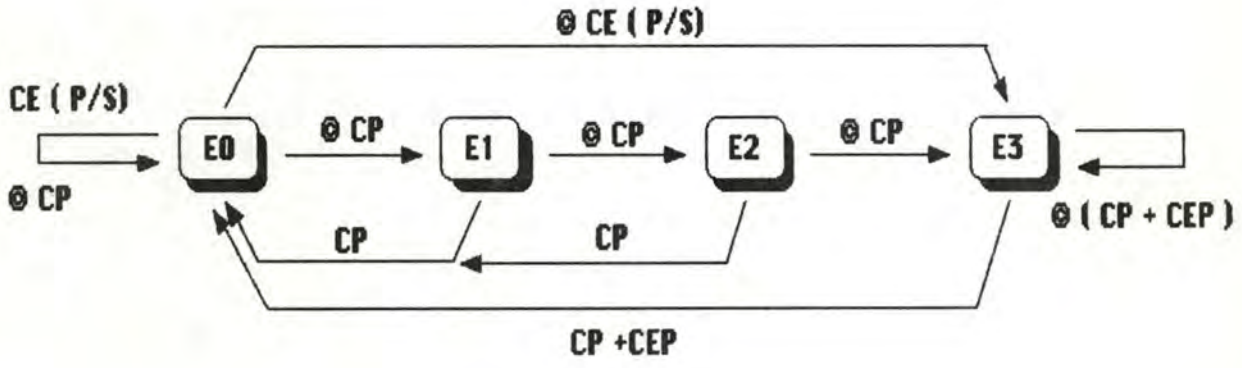


Fig. 3.6 : Les états de la carte

Les états sont les suivants :

E0 : état non bloqué, fonctionnement normal
E1 : état après 1 erreur
E2 : état après 2 erreurs
E3 : état bloqué

Les présentations de clés possibles sont les suivantes :

CP	: présentation d'une clé porteur correcte
CE	: présentation d'une clé émetteur correcte
©CP	: présentation d'une clé porteur incorrecte
©CE	: présentation d'une clé émetteur incorrecte
CP + CEP	: recyclage avec de bonnes clés porteur et émetteur primaire
©(CP + CEP)	: recyclage avec de mauvaises clés porteur ou émetteur primaire

N.B. P/S : primaire ou secondaire.

c) La zone confidentielle

La zone confidentielle contient des informations confidentielles propres à l'application. Celles-ci sont non évolutives.

d) La zone de transactions

La zone de transactions contient les autres informations propres à l'application et inscrites dans la carte à partir de sa mise en service. C'est en principe la plus grande zone de la mémoire PROM.

Y sont inscrits, par exemple, la liste des transactions effectuées par le porteur, les blocs de bits de prépaiement, ou encore les paramètres de connexion automatique à des services proposés sur des serveurs tels que le minitel.

e) La zone de lecture

La zone de lecture est semblable à la zone confidentielle. Cependant les informations qui y sont stockées (n° de compte du porteur, adresse ...), sont

accessibles à toute personne. Leur lecture ne nécessite pas de présentation de clé.

f) La zone de fabrication

La zone de fabrication contient :

- les informations relatives à la taille des zones et à leur emplacement,
- les informations techniques relatives à la puce (n° de série , ...).

g) La zone des verrous

La zone des verrous signale la phase du cycle de vie dans laquelle la carte se trouve. Les différentes phases de vie seront détaillées au point 3.4. (La vie de la carte).

Sur la figure 3.7, nous trouvons le **contenu des 7 zones** :

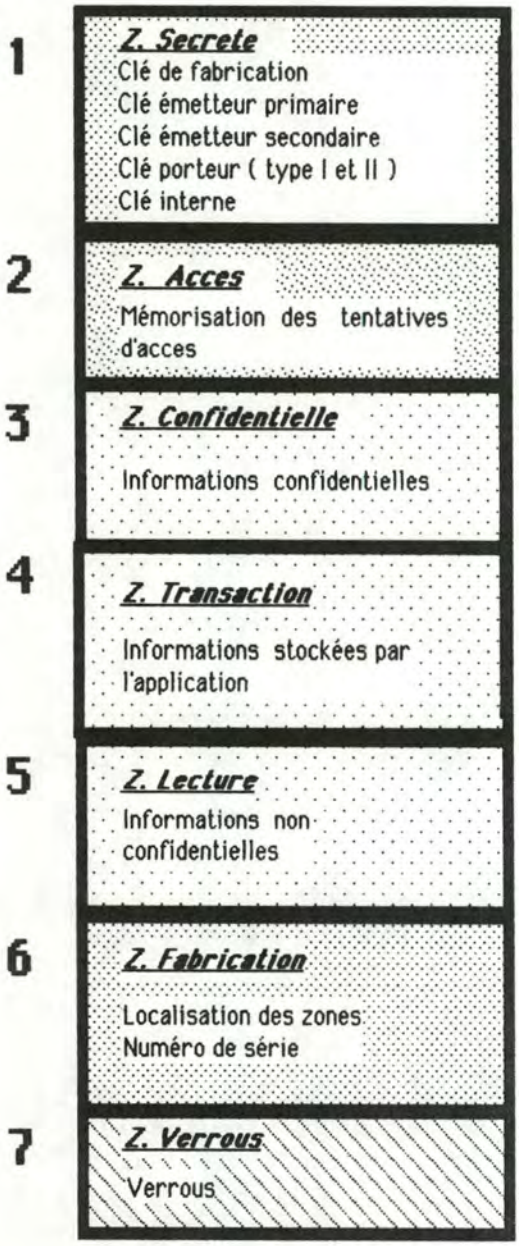


Fig. 3.7 : Le contenu des 7 zones

3.2.1.2. Les mots

Un mot compte 32 bits (fig. 3.8).

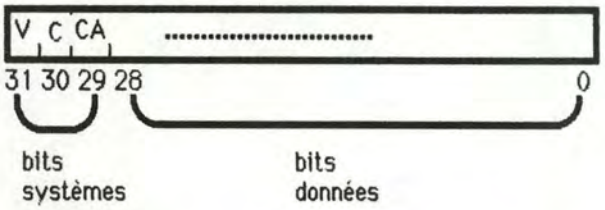


Fig. 3.8 : Le contenu d'un mot mémoire

Les 29 bits de poids faible sont les bits de données.
Les 3 bits de poids fort sont des bits systèmes.

Les bits C et CA spécifient le type d'utilisateur qui a écrit le mot (fig. 3.9).

Ces bits permettent, par exemple, au programme d'application de vérifier que les modalités d'utilisation de la carte pour un service donné ont bien été écrites par l'émetteur concerné ou que les transactions ont été réalisées par le porteur.

C	CA	
1	1	Mot écrit par l'émetteur primaire
1	0	Mot écrit par l'émetteur secondaire
0	X	Mot écrit par le porteur

Fig. 3.9 : Les bits C et CA

Le bit V permet de signaler qu'un mot est valide ou non.

Pour qu'un mot soit validé, la clé correspondant aux bits C et CA doit avoir été été présentée. Par exemple, un mot ayant 1 et 1 pour valeurs respectives des bits C et CA ne sera validé que si l'utilisateur a présenté au préalable la clé émetteur primaire.

Un mot validé ne peut plus jamais être modifié.

3.2.2. Fonction logico-mathématique

Nous avons déjà signalé qu'en matière de sécurité de l'information, il est nécessaire de protéger les informations lors de leur communication ou de leur stockage; d'où l'utilité des algorithmes de chiffrement et bien sûr de déchiffrement.

Les cryptosystèmes sont des algorithmes qui ont pour but de transformer une information afin qu'elle devienne inintelligible et donc inutile pour ceux à qui elle n'est pas destinée.

On distingue trois grandes catégories de cryptosytèmes :

- Algorithme secret à clé secrète : l'algorithme doit être conservé secret ainsi que la clé de chiffrement.

- Algorithme public à clé secrète : l'algorithme peut être ici connu de tous. Seule la clé doit rester secrète .

- Algorithme public à clé révélée : l'algorithme, ainsi que la clé de chiffrement sont divulgués. Seule la clé de déchiffrement reste secrète. L'émetteur peut donc transmettre un message chiffré au récepteur qui, seul, possède la clé de déchiffrement. La clé de chiffrement (publique) est donc différente de celle de déchiffrement (secrète).

La fonction logico-mathématique de la carte est un cryptosystème développé par BULL CP8, couramment appelé algorithme Télépass. Télépass se situe dans la catégorie des algorithmes secrets à clé secrète. Il est implémenté sur la carte à microprocesseur et c'est la clé interne de celle-ci qui joue le rôle de clé secrète dans l'algorithme Télépass. A aucun moment, l'algorithme ou sa clé secrète n'est accessible de l'extérieur .

La fonction est de la forme suivante :

R = F (E , S , adr carte , (adr carte))

- R est le résultat de la fonction, appelé aussi certificat.

- E est une information fournie par l'extérieur appelée message d'entrée.

- S est la clé interne de la carte. Etant donné que cette information est secrète, elle garantit à R un caractère inimitable.

- **adr carte** est l'adresse d'un mot situé en mémoire PROM de la carte.

- **(adr carte)** est le contenu du mot dont l'adresse est adr carte (mot d'une mémoire protégée ou non). Ce mot est généralement appelé paramètre interne de la carte .

Le schéma de la figure 3.10 illustre cette fonction Télépass .

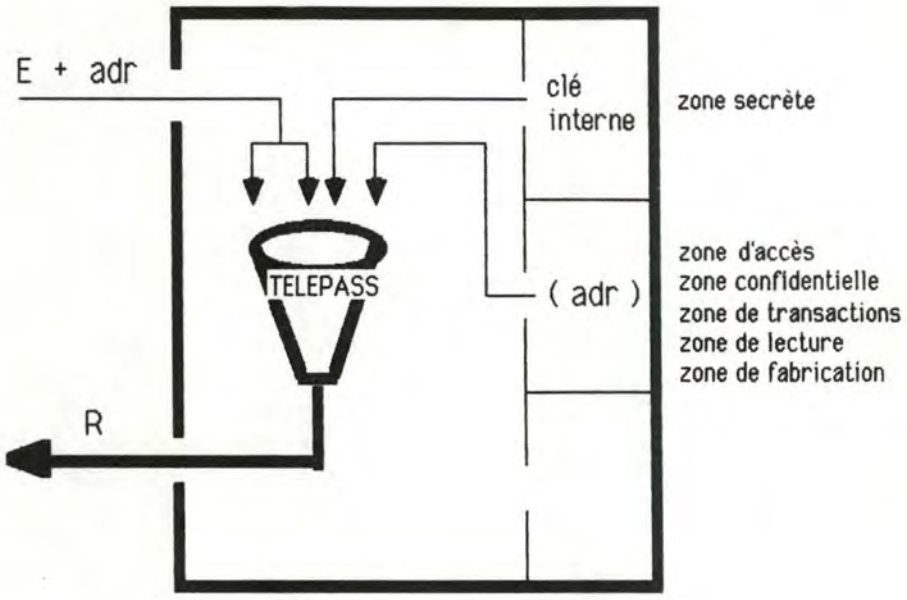


Fig. 3.10 : Schéma de Télépass

Cette fonction F n'est pas réversible. En d'autres termes, il est impossible de déduire de la forme de F une des trois formes ci-dessous :

F' tel que F' (S , R , adr , (adr)) ---> E

F'' tel que F'' (E , R , adr , (adr)) ---> S

F''' tel que F''' (S , R , E) ---> adr, (adr)

Cela signifie que Télépass est un algorithme de chiffrement pour lequel il n'existe pas d'algorithme de déchiffrement correspondant.

L' algorithme Télépass est utilisé pour sécuriser le dialogue entre la carte et un système informatique externe (terminal, ordinateur central, ...). Ce système informatique doit alors posséder un module sécuritaire comprenant un copie de l'algorithme et connaissant le secret S utilisé par la carte dans le dialogue.

3.2.3. Gestion des accès à la mémoire PROM

Le microprocesseur est présent pour s'interposer en permanence entre le monde extérieur et la mémoire PROM. Cela signifie que toute action sur la mémoire PROM nécessite de la part du monde extérieur un ordre envoyé au microprocesseur .

Deux actions sur la mémoire sont autorisées : l'écriture et la lecture (la modification et l'effacement étant impossibles).

Les règles d'accessibilité pour le microprocesseur sont différentes des règles d'accessibilité pour les programmes d'application.

3.2.3.1. Lecture et écriture par le microprocesseur

Le microprocesseur peut accéder aux zones de la mémoire PROM de la façon suivante :

	ACCESSIBILITE	
	EN LECTURE	EN ECRITURE
Zone secrète	oui	oui (1)
Zone d'accès	oui	oui
Zone confidentielle	oui	non
Zone de transactions	oui	oui
Zone de lecture	oui	non
Zone de fabrication	oui	non
Zone des verrous	oui	oui

Fig. 3.11 : Droit d'accès du microprocesseur

(1) : L'écriture dans la zone secrète ne peut se faire que dans le cas d'un changement de la clé porteur.

Comme la figure 3.11 le montre, seule l'écriture en zone confidentielle, en zone de lecture et en zone de fabrication est interdite au microprocesseur. La raison en est simple : après la mise en service de la carte, les trois zones citées précédemment sont déjà entièrement remplies.

3.2.3.2. Lecture et écriture de l'extérieur

Les programmes d'application disposent des possibilités suivantes :

	ACCESSIBILITE	
	EN LECTURE	EN ECRITURE
Zone secrète	non (2)	oui (1)
Zone d'accès	oui	non
Zone confidentielle	oui (2)	non
Zone de transactions	oui (2)(3)	oui (2)(3)
Zone de lecture	oui	non
Zone de fabrication	oui	non
Zone des verrous	oui	oui

Fig. 3.12 : Droit d'accès de l'extérieur

oui : La zone est totalement accessible sans présentation préalable de clé . Elle est dite accessible et libre .
non : La zone est totalement inaccessible .
(1) : L'écriture dans la zone secrète ne peut se faire que pour un changement de clé porteur.
(2) : La zone est accessible mais protégée c'est-à-dire que l'accès ne peut se faire qu'après présentation d'une clé porteur ou émetteur.
(3) : La zone de transactions peut être protégée ou non selon le désir de l'émetteur .

Nous remarquons en comparant ces deux tableaux que le microprocesseur peut réaliser deux actions interdites aux programmes d'application :

- lecture dans la zone secrète : afin de pouvoir comparer la clé présentée par l'extérieur avec la clé secrète correspondante, ou pour utiliser la clé interne dans la fonction TELEPASS.

- écriture dans la zone d'accès : afin de comptabiliser les tentatives d'accès de l'extérieur .

3.2.4. Jeu d'instructions du microprocesseur

Les instructions sont de trois types :

- instructions d'initialisation,
- instructions basées sur l'algorithme Télépass,
- instructions simples.

Les instructions d'initialisation et les instructions simples renvoient généralement deux octets qui signalent comment l'instruction s'est déroulée et qui décrivent l'état de la carte à ce moment.

3.2.4.1. Instructions d'initialisation

- La seule instruction de cette catégorie est la **remise à zéro**. Son rôle est de préparer le dialogue, c'est-à-dire initialiser la carte puis renvoyer au lecteur-encodeur de cartes les paramètres décrivant la carte insérée (conformément aux normes ISO).

3.2.4.2 Instructions basées sur l'algorithme Télépass

- Activation de la **fonction Telepass** : cette instruction calcule un certificat (R) au départ des arguments qui lui sont fournis (message d'entrée (E) , adresse du paramètre interne (adr) et des informations qui se trouvent dans la carte (clé interne (S) et paramètre interne (mot pointé par adr)).

L'adresse du paramètre interne doit être dans une zone accessible en lecture. Si cette zone est protégée en lecture, il est nécessaire d'avoir préalablement présenté et validé la bonne clé en lecture.

3.2.4.3. Instructions simples

- **Présentation d'une clé** : la présentation d'une clé est réalisée par trois ordres différents en fonction de la clé à présenter :

  - présentation d'une clé porteur ou d'une clé de fabrication,
  - présentation d'une clé émetteur primaire,
  - présentation d'une clé émetteur secondaire.

Ces ordres ont pour but de fournir une clé à la carte. La carte peut alors comparer la clé reçue avec celle présente en zone secrète. Le résultat de cette comparaison n'est pas fourni à l'extérieur .

- **Validation de clé en lecture** : cette instruction réalise la mise à jour de la zone d'accès à la suite d'une présentation de clé. Cette instruction doit être exécutée avant toute lecture et pour tout recyclage.

- **Lecture** : cette instruction a pour but de lire un nombre donné d'octets à partir d'une adresse donnée.

- **Ecriture d'un mot (32 bits)** : cette instruction réalise l'écriture d'un mot dans une zone protégée ou non en écriture . On ne peut écrire sur un mot validé, mais on peut réécrire sur un mot déjà utilisé mais non validé (par exemple pour écrire un seul bit).

- **Validation en écriture** : cet ordre réalise la validation d'un mot pour autant que les conditions de validation (définies au point 3.2.1.2.) soient remplies.

- **Lecture du résultat** : cette instruction retourne au monde extérieur le résultat R calculé par l'instruction d'activation de la fonction Télépass.

- **Ecriture des verrous** : cette instruction marque la fin d'une des phases de la vie de la carte en positionnant le verrou correspondant. Elle est aussi utilisée afin d'invalider une carte (l'invalidation sera définie au point 3.4.1.5).

- **Ecriture d'un mot de test** : pour mémoire.

3.2.5. Gestion des échanges entre la carte et le lecteur-encodeur

Le protocole adopté pour la gestion des échanges est conforme aux normes ISO sur les cartes à microprocesseur. Pour tout renseignement, se référer aux normes 7816/3 et 7816/3 Addendum1 en annexe 1.

3.3. Les programmes d'application

3.3.1. Environnement d'exécution

Les services accessibles au moyen de la carte à microprocesseur sont informatisés. Il doit donc exister des programmes d'application réalisant la mise en oeuvre de ces services.

De toute évidence, la carte ne possède pas une mémoire suffisante pour permettre le stockage de ces programmes d'application. De plus, ces derniers réalisent souvent des fonctions complémentaires à la gestion de la carte (par exemple, la mise à jour du compte client après une transaction). Les programmes d'application sont donc stockés et exécutés sur un système externe. En fonction de l'application , le système externe peut être :

- le lecteur-encodeur qui reçoit la carte (ces appareils sont généralement dotés d'un processeur et d'une mémoire d'environ 32 Koctets),

- un micro-ordinateur relié directement à un lecteur-encodeur,

- un gros système relié à une multitude de lecteurs-encodeurs via un réseau de télécommunications.

La figure 3.13 illustre le système externe constitué d'un micro-ordinateur et d'un lecteur-encodeur

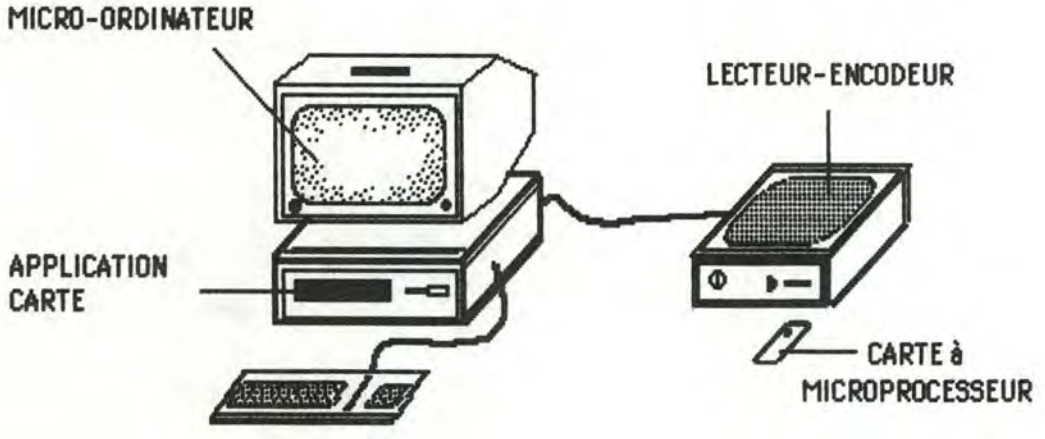


Fig. 3.13 : Un système externe

3.3.2. Principe des communications avec la carte

Pour les programmes d'application, la carte peut être perçue comme un moyen de stockage, au même titre que les disques ou les bandes magnétiques, et le microprocesseur peut être considéré comme un processeur d'entrées/sorties. Toutefois, à cause de la spécificité du support, les langages de programmation ne disposent pas de primitives standardisées permettant une manipulation aisée de la carte à microprocesseur.

Pour assurer la gestion de la carte, les programmes d'application ne peuvent utiliser que les seules instructions directement exécutables par le microprocesseur (cfr 3.2.4. Jeu d'instructions du microprocesseur).

En conséquence, les communications entre la carte et les systèmes externes sont constituées exclusivement de flots d'instructions appartenant au jeu d'instructions du microprocesseur et de leurs valeurs de retour.

3.3.3. Exemples typiques de communication avec la carte

Les exemples les plus typiques de communication entre la carte et les programmes d'application - que nous allons détailler ici - sont les opérations de lecture et d'écriture. Nous verrons d'autres exemples de communication dans le chapitre 4 lorsque nous aborderons la réalisation des fonctions de sécurité.

Les opérations de lecture et d'écriture diffèrent lorsqu'elles sont réalisées sur des zones libres ou sur des zones protégées.

a) Lecture dans une zone libre

Pour réaliser une lecture en zone libre, il suffit d'utiliser l' **ordre de lecture** avec ses paramètres (adresse de début de lecture et nombre d'octets à lire).

Les zones libres en lecture sont la zone de lecture, la zone de fabrication, la zone des verrous et, éventuellement, la zone de transactions.

b) Ecriture dans une zone libre

L'écriture dans une zone libre comporte deux étapes :

- L'envoi à la carte d'un **ordre d'écriture** et de ses paramètres (adresse de l'écriture et mot à écrire).

- L'envoi d'un ordre de **validation d'écriture** avec, comme paramètre, l'adresse du mot à valider.

Les zones libres en écriture sont la zone des verrous et, éventuellement, la zone de transactions.

Les lectures et les écritures en zone libre sont illustrées à la figure 3.14 :

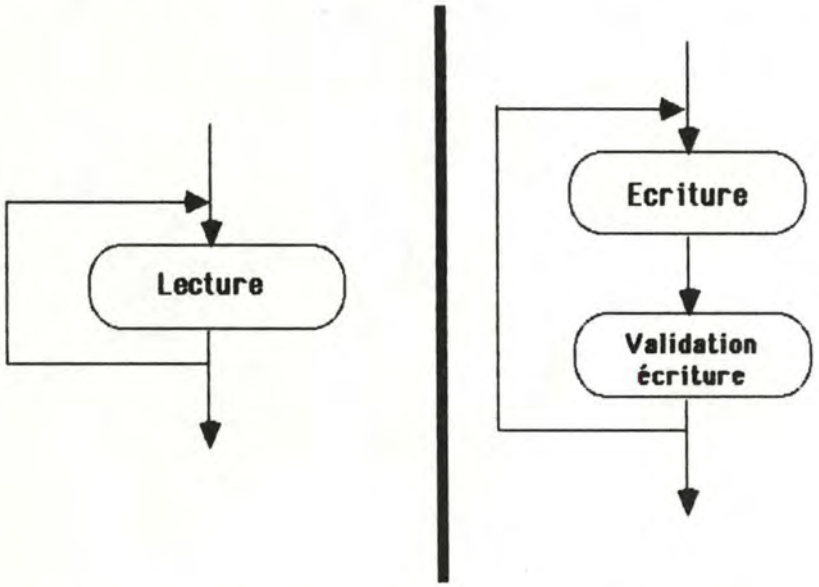


Fig. 3.14 : Scénario de lecture et d'écriture dans une zone non protégée

c) Lecture dans une zone protégée

Une lecture dans une zone protégée comporte trois étapes :

- La **présentation d'une clé** à la carte : soit une clé porteur, soit une clé émetteur.

- Une demande de **validation de cette clé en lecture**. Si la clé est correcte, toutes les lectures dans les zones protégées par la clé concernée seront autorisées. Dans le cas contraire, la carte comptabilise cette mauvaise présentation de clé dans la zone d'accès .

- L'envoi vers la carte d'un **ordre de lecture** et de ses paramètres .

Les zones protégées en lecture sont la zone d'accès, la zone confidentielle et, éventuellement, la zone de transactions.

d) Ecriture dans une zone protégée

Pour réaliser l'écriture dans une zone protégée , il est aussi nécessaire de passer par trois phases qui sont sensiblement les mêmes que celles vues pour la lecture protégée :

- La **présentation d'une clé** à la carte.

- L'envoi vers la carte d'un **ordre d'écriture** et de ses paramètres (adresse d'écriture et mot à écrire) .

- Une demande de **validation d'écriture** pour le mot qui vient d'être écrit.

Contrairement à la validation en lecture qui a lieu avant la lecture, la validation en écriture doit avoir lieu après l'écriture afin de pouvoir vérifier que les bits C et CA correspondent à la clé présentée.

La seule zone qui peut être protégée en écriture est la zone de transactions.

Les lectures et les écritures dans une zone protégée sont illustrées à la figure 3.15 :

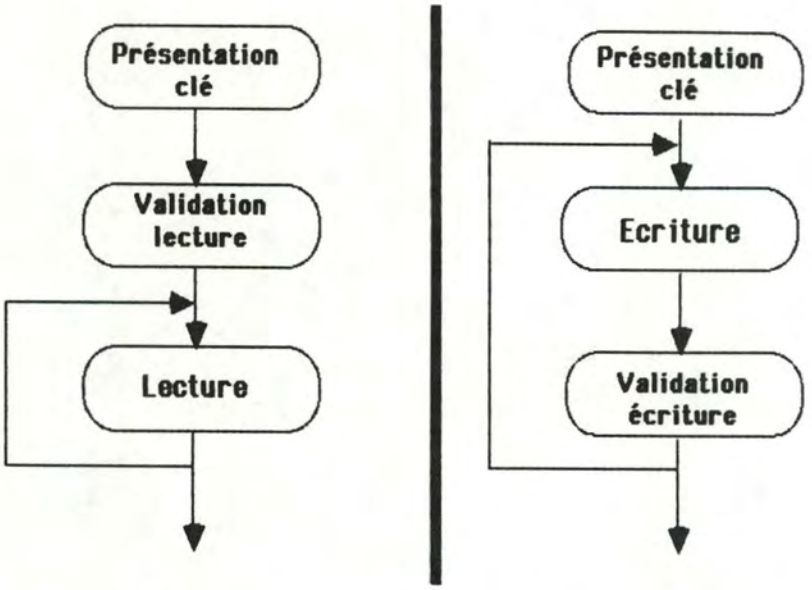


Fig. 3.15 : Scénario de lecture et d'écriture dans une zone protégée

3.4. Cycle de vie de la carte

3.4.1. Les phases de la vie d'une carte

La vie de la carte se schématise de la façon suivante :

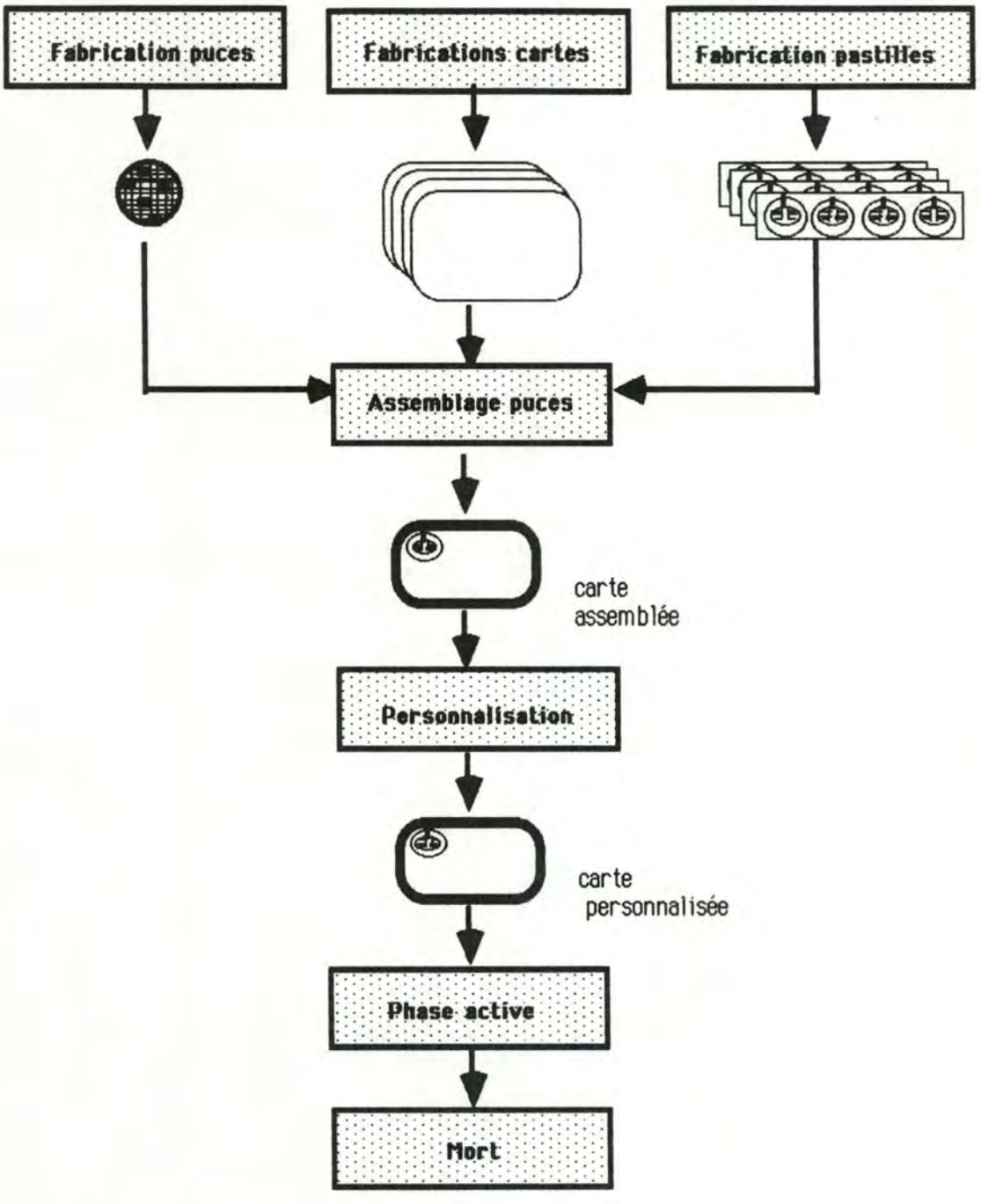


Fig. 3.16 : Vie de la carte

La vie de la carte englobe la fabrication de ses différents composants, leur assemblage, la personnalisation (c'est-à-dire l'initialisation des mémoires), la vie active (c'est-à-dire son utilisation pour des services divers) et se termine par la mort de la carte.

3.4.1.1. La phase de fabrication

La fabrication d'une carte comprend celle des puces, celle des pastilles et celle des cartes plastiques. Voyons de plus près la fabrication des puces. Durant cette phase :

- Le constructeur fabrique le microprocesseur, la ROM, la RAM et la PROM.

- Il écrit le masque, c'est-à-dire les programmes de la ROM.

- Il crée, dans la mémoire PROM, la zone de fabrication et celle des verrous.

- Il inscrit le numéro d'identification de la carte dans la zone de fabrication et la clé de fabrication dans la zone secrète. Il est important de noter que chaque carte du lot de fabrication possède déjà une clé de fabrication unique.

- On positionne, dans la zone des verrous, le verrou LF. Cela indique que la phase de fabrication est terminée et que les règles d'accessibilité à la mémoire PROM sont les suivantes :

    - lecture : libre, sauf pour la clé de fabrication.
    - écriture : protégée (aucune écriture ne peut être tentée sans présentation préalable de la clé de fabrication).

Le lot de puces est ainsi fabriqué puis envoyé vers l'endroit d'assemblage.

3.4.1.2. La phase d'assemblage

La phase d'assemblage comprend :

- le test des puces,
- l'encartage de la puce et de la pastille dans la carte plastique,
- l'écriture d'un numéro de série.

3.4.1.3. La phase de personnalisation

La personnalisation consiste à inscrire dans la mémoire PROM les informations caractéristiques des services proposés ainsi que les données afférentes au destinataire final de la carte (le porteur).

Plus précisément, la personnalisation comprend :

- l'écriture des informations définissant la taille, la localisation des différentes zones (secrète, accès, confidentielle, transactions, lecture),

- l'écriture du type de protection en lecture et en écriture de la zone de transactions,

- l'écriture dans les zones appropriées des informations suivantes :

    - données sur le futur porteur de la carte (identité bancaire, adresse),
    - données relatives au service offert par la carte,
    - les clés secrètes : celle du porteur , du (des) émetteur(s) et enfin la clé interne .

- le positionnement du verrou indiquant la terminaison de la phase de personnalisation (la clé de fabrication est alors désactivée tandis que les clés porteur, émetteur et interne sont activées).

Cette phase est réalisée par l'emetteur des cartes qui est le seul à connaître la clé de fabrication de chacune des cartes.

A partir de la fin de la personnalisation, les règles d'accessibilité à la mémoire PROM sont celles définies au point 3.2.3 (Gestion des accès à la mémoire PROM .

La carte est alors envoyée à chaque porteur. La clé porteur leur est communiquée par une autre voie.

Remarque : la personnalisation des cartes est réalisée par des programmes analogues aux programmes d'application.

3.4.1.4. La phase active

La phase active correspond à la phase d'utilisation de la carte par son porteur.

Les actions réalisables par le porteur dépendent des services en fonction desquels la carte a été personnalisée.

A titre indicatif, citons les actions les plus courantes :

- effectuer des transactions,
- visualiser la liste des transactions déjà réalisées,
- changer la valeur de la clé porteur (la première clé devient alors inactive),
- débloquer une carte.

3.4.1.5. La mort de la carte

La mort de la carte peut survenir dans quatre cas :

- saturation de la zone d'accès,
- saturation de la zone de transactions,
- invalidation de la carte par le programme d'application (par exemple lorsque la carte insérée figure sur une liste noire),
- auto-invalidation de la carte dès qu'une condition anormale d'utilisation est détectée (par exemple lorsque la mémoire PROM a été effacée).

Remarque : invalider une carte consiste à inhiber certaines instructions du microprocesseur. L'invalidation est réalisée par le positionnement d'un verrou dans la zone des verrous. Lors de toute mise sous tension de la carte , le verrou d'invalidation est testé. Si ce dernier est positionné, le microprocesseur refusera d'exécuter les instructions inhibées.

3.4.2. Protection de la carte pendant le cycle de vie

La carte est protégée contre une utilisation illégitime durant toutes les étapes du cycle de vie.

Les sites de fabrication, d'assemblage et de personnalisation étant souvent différents, la puce est transmise de l'usine de fabrication vers le porteur en passant par l'usine d'assemblage et par l'émetteur. Il faut donc se prémunir contre les vols de cartes et, le cas échéant, éviter que le voleur puisse ensuite utiliser les cartes volées tout à fait normalement .

C'est pourquoi, dès la fin de la fabrication de la puce, une clé est inscrite dans la carte **(clé de fabrication)**. Dès ce moment, on ne peut plus écrire sur la carte sans avoir au préalable présenté cette clé de fabrication. Il est donc impossible pour un fraudeur de personnaliser une carte volée après la fabrication.

Lorsqu'une carte a été personnalisée, la clé de fabrication est désactivée au profit des autres clés contenues dans la zone secrète (clé émetteur, clé porteur, clé interne). Ce sont alors ces nouvelles clés qui préviennent toute utilisation abusive de la carte.

IV. EVALUATION DE LA CARTE CP8

Dans ce chapitre, nous allons voir dans quelles mesures la carte décrite au chapitre trois satisfait aux critères d'évaluation vus au chapitre deux (protection de l'information, polyvalence et aspect multi-services).

4.1. Protection de l'information

Avant de voir en détail la protection de l'information mise en oeuvre grâce à la carte, nous tenons à faire remarquer que le niveau de protection atteint est très élevé. Dans notre étude, nous n'avons relevé qu'une seule faiblesse que nous signalerons en temps opportun.

4.1.1. Sécurité physique

Les dispositifs de sécurité implémentés au niveau physique découlent de la **nature et de l'architecture de la puce**. Les protections sont les suivantes:

- La carte résiste aux champs électromagnétiques, aux rayons UV et aux rayons X inférieurs à 20000 rads.

- La technologie utilisée pour la fabrication des puces (technologie FAMOS) garantit que les mémoires ne sont lisibles que via le bus des données et celui des adresses.

- La structure monolithique de la puce empêche tout accès aux bus qui ne serait pas autorisé par le microprocesseur (à l'exception des accès via les plots de test).

- Les plots de tests qui permettent un accès direct aux bus internes et qui sont utilisés pour tester les mémoires avant la mise en service des cartes sont détruits à la fin de la phase de fabrication.

- L'effacement global de la mémoire EPROM, qui est théoriquement possible par rayons UV, est contrôlé par le microprocesseur. Ce dernier peut détecter un effacement de sa mémoire de stockage en testant la valeur de certains bits appelés témoins d'effacement. En cas d'effacement, le microprocesseur peut rendre la carte inutilisable.

- En cas d'utilisation anormale (chute de tension de l'écriture, ralentissement de l'horloge, ...), le microprocesseur se bloque.

- Les plans de fabrication de la puce sont secrets. Cela rend la duplication d'une carte très peu probable car cette duplication nécessiterait des compétences et des investissements considérables.

4.1.2. Sécurité logique

La sécurité logique repose sur les cinq fonctions de sécurité vues au chapitre deux. Toutefois, il ne suffit pas d'implémenter ces cinq fonctions pour garantir un haut niveau de sécurité logique. En effet, ces cinq fonctions sont basées sur l'utilisation de clés secrètes, et la divulgation de ces clés rendrait les fonctions de sécurité inutiles. Pour garantir un haut degré de sécurité logique, il faut donc :

- implémenter les fonctions de sécurité,
- assurer la confidentialité des clés.

4.1.2.1. Implémentation des fonctions de sécurité

a) Authentification de l'utilisateur

La figure 4.1 illustre le principe de l'authentification de l'utilisateur.

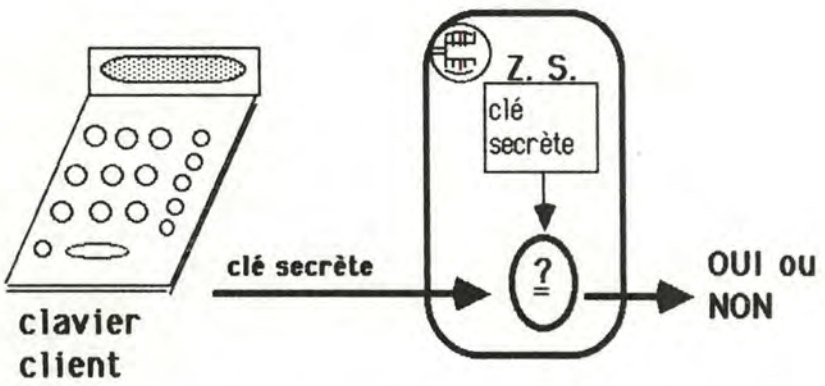


Fig 4.1 : L'authentification de l'utilisateur

Le principe d'authentification de l'utilisateur est simple : après avoir été saisie au clavier, la clé secrète (clé porteur, clé émetteur ou clé de fabrication) est envoyée à la carte qui la compare avec celle stockée dans la zone secrète.

Cette fonction est réalisée par l'instruction de **présentation de clé**.

b) Authentification de la carte

Lorsqu'un système externe veut s'assurer de l'authenticité d'une carte, il lui demande de présenter sa clé interne.

Rappelons qu'un système externe possède un module sécuritaire qui dispose de l'algorithme Télépass et qui connaît la clé interne de chaque carte avec laquelle le système externe est susceptible de dialoguer.

L'authentification de la carte est réalisée par le programme d'application du système externe de la manière suivante :

- Le programme d'application génère un nombre aléatoire E.

- Le programme d'application demande à la carte d'**activer la fonction Télépass** et lui fournit comme arguments le nombre aléatoire E et l'adresse ADR d'un mot situé dans la mémoire de stockage de la carte.

- Le programme d'application envoie à la carte l'ordre de **lecture du résultat** et reçoit en retour un certificat R.

- Le programme d'application réalise une **lecture** du mot situé à l'adresse ADR (parfois cette étape peut être évitée si le programme d'application connaît à priori le contenu de ce mot).

- Le programme d'application calcule un certificat R' en prenant comme paramètres :

  - le nombre aléatoire E,
  - l'adresse ADR,
  - le contenu (ADR) du mot situé à l'adresse ADR,
  - la clé interne de la carte à authentifier.

- Le programme d'application compare les certificats R et R'. S'ils concordent, cela signifie que la clé interne de la carte est correcte, et donc que la carte est authentique.

La figure 4.2 illustre le principe d'authentification d'une carte.

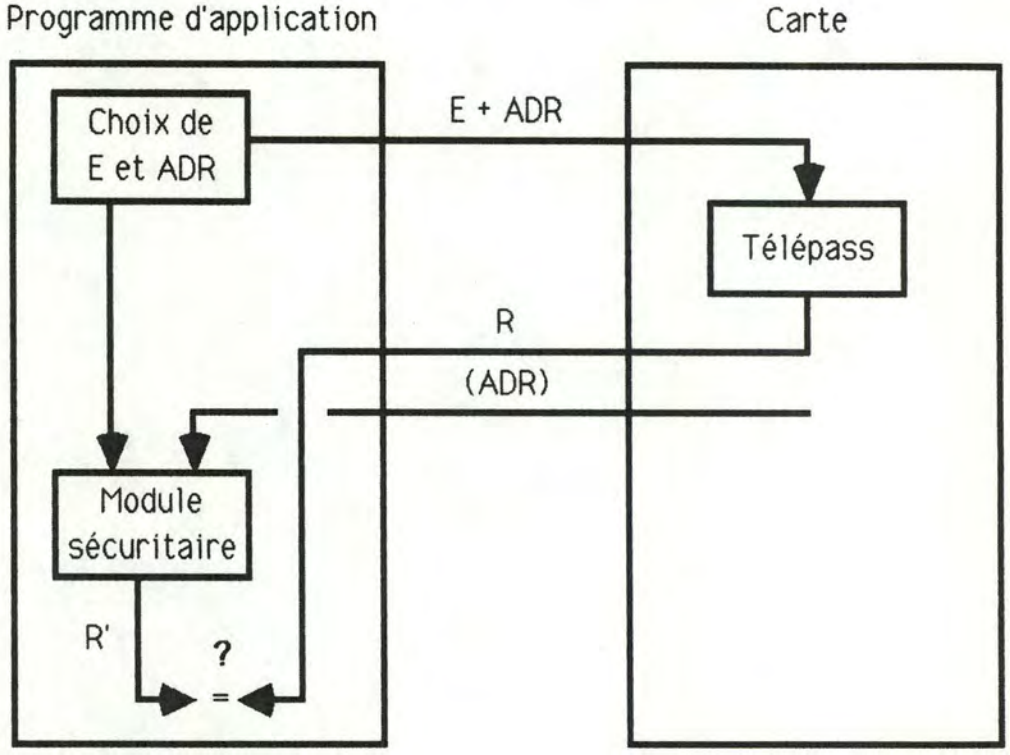


Fig 4.2 : L'authentification d'une carte

c) La certification

Rappelons que la certification consiste à fournir un certificat relatif à une information déterminée et que le certificat sert à garantir la présence de cette information dans la carte.

La certification est réalisée sur un ordre du programme d'application du système externe de la manière suivante :

- Le programme d'application génère un nombre aléatoire E.

- Le programme d'application demande à la carte d'**activer la fonction Télépass** et lui fournit comme arguments le nombre aléatoire E et l'adresse ADR du mot pour lequel le système externe veut un certificat.

- Le programme d'application envoie à la carte l'ordre de **lecture du résultat** et reçoit en retour un certificat R.

- Le programme d'application calcule un certificat R' en prenant comme paramètres :

    - le nombre aléatoire E,
    - l'adresse ADR,
    - le contenu (ADR) du mot situé à l'adresse ADR (ce mot doit être connu à priori par le programme d'application),
    - la clé interne de la carte.

- Le programme d'application compare les certificats R et R'. S'ils concordent, cela signifie que le mot situé à l'adresse ADR est effectivement celui attendu par le programme d'application.

La figure 4.3 illustre le principe de la certification.

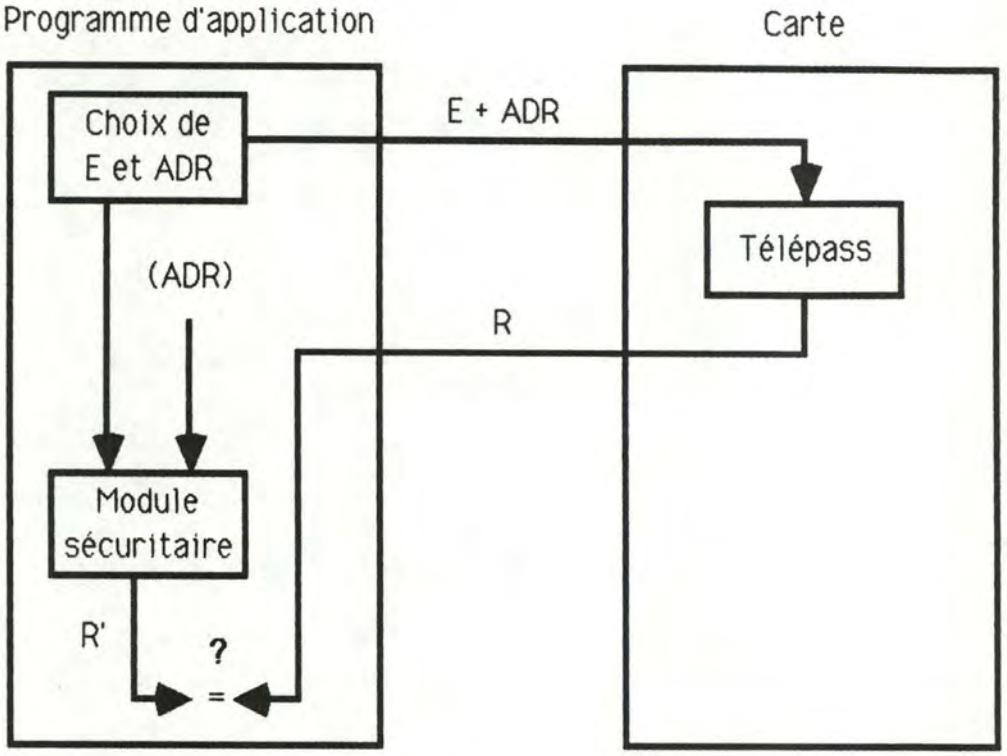


Fig 4.3 : La certification

Remarque : à première vue, on pourrait penser qu'une simple lecture du mot situé à l'adresse ADR est équivalente à une certification. Il n'en est rien! En effet, si la lecture permet de garantir l'existence de l'information, rien ne prouve que cette information se trouve sur une carte déterminée car la lecture ne fait pas référence à la clé interne de la carte.

d) Signature électronique

Rappelons que la signature électronique est utilisée pour vérifier l'authenticité et l'absence d'altérations d'un message au cours d'une transmission.

Pour cette fonction, le dialogue ne s'établit plus entre un programme d'application et une carte comme précédemment, mais entre un programme émetteur et un programme récepteur. Ces deux programmes doivent disposer respectivement d'une carte CP8 et d'un module sécuritaire.

Le principe de la signature électronique est le suivant :

- L'émetteur condense le message M à transmettre en une chaîne de six octets désignée par E.

- L'émetteur génère un certificat R en utilisant comme paramètres :

  - le message condensé E,
  - l'adresse ADR d'un mot situé dans la mémoire de stockage de la carte (le contenu de ce mot indique généralement l'identité de l'émetteur),
  - le contenu (ADR) du mot dont l'adresse est ADR,
  - la clé interne de la carte utilisée pour le dialogue.

- L'émetteur envoie au récepteur :

  - le message M,
  - le certificat R,
  - l'adresse ADR,
  - éventuellement le contenu (ADR) du mot situé à l'adresse ADR.

- Le récepteur condense le message reçu M' en une chaîne de six octets désignée par E'.

- Le récepteur génère un certificat R' en utilisant comme paramètres :

  - le message condensé E',
  - l'adresse ADR,
  - le contenu (ADR) du mot dont l'adresse est ADR,
  - la clé interne de la carte.

- Le récepteur compare les certificats R et R'. S'ils concordent, les messages M et M' sont identiques.

La figure 4.4 résume le principe de la signature électronique.

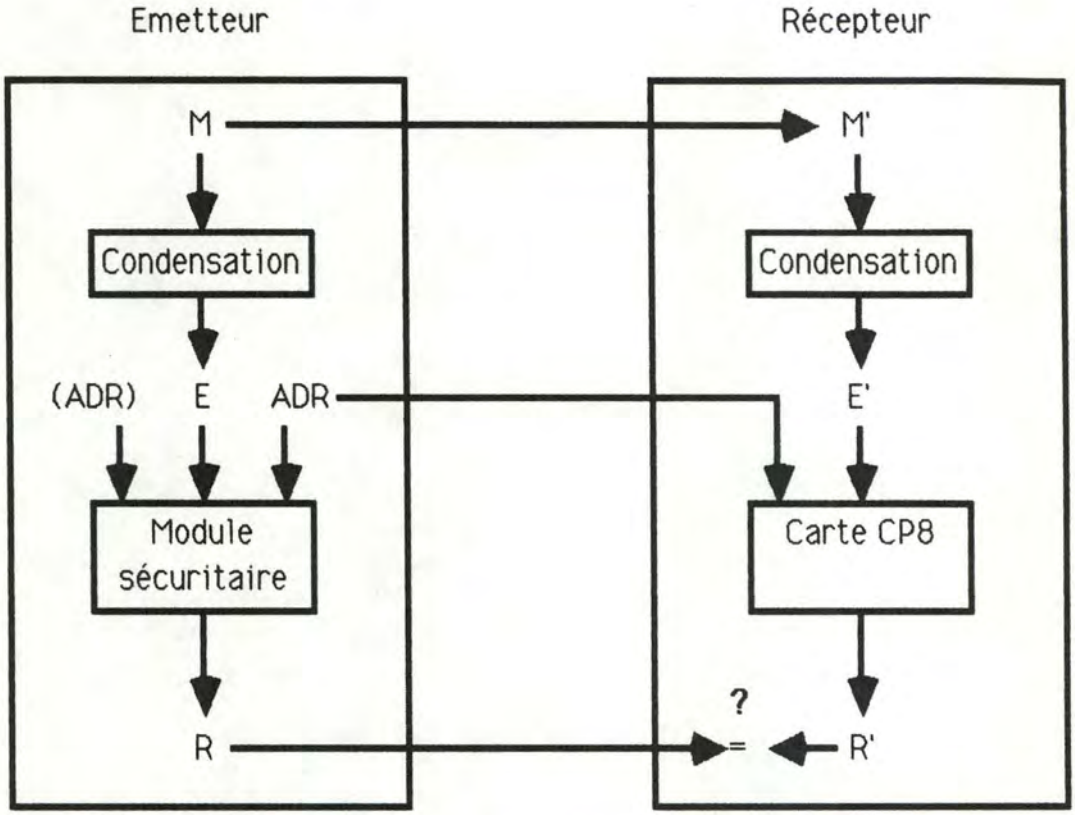


Fig 4.4 : La Signature électronique

Remarque : la signature électronique pourrait être réalisée de la même manière si l'émetteur disposait du module sécuritaire et si le récepteur disposait de la carte CP8.

e) Génération de clés de chiffrement

Pour réaliser la génération de clés de chiffrement, les programmes de chiffrement et de déchiffrement doivent disposer respectivement d'un module sécuritaire et d'une carte CP8.

Le principe de la génération de clés de chiffrement est le suivant :

- Le programme de chiffrement génère un nombre aléatoire E.

- Le programme de chiffrement envoie au programme de déchiffrement le nombre aléatoire E et l'adresse ADR d'un mot situé dans la mémoire de stockage de la carte.

- Le programme de chiffrement réalise une **lecture** du mot situé à l'adresse ADR (parfois cette étape peut être évitée si le programme de chiffrement connaît à priori le contenu de ce mot).

- Le programme de chiffrement calcule un certificat R en prenant comme paramètres :

    - le nombre aléatoire E,
    - l'adresse ADR,
    - le contenu (ADR) du mot situé à l'adresse ADR,
    - la clé interne de la carte.

- Le programme de déchiffrement calcule un certificat R' en prenant comme paramètres :

    - le nombre aléatoire E,
    - l'adresse ADR,
    - le contenu (ADR) du mot situé à l'adresse ADR,
    - la clé interne de la carte.

- Sauf erreur de transmission, les certificats R et R' sont identiques et constituent une clé de chiffrement.

La figure 4.5 illustre le principe de la génération de clés de chiffrement.

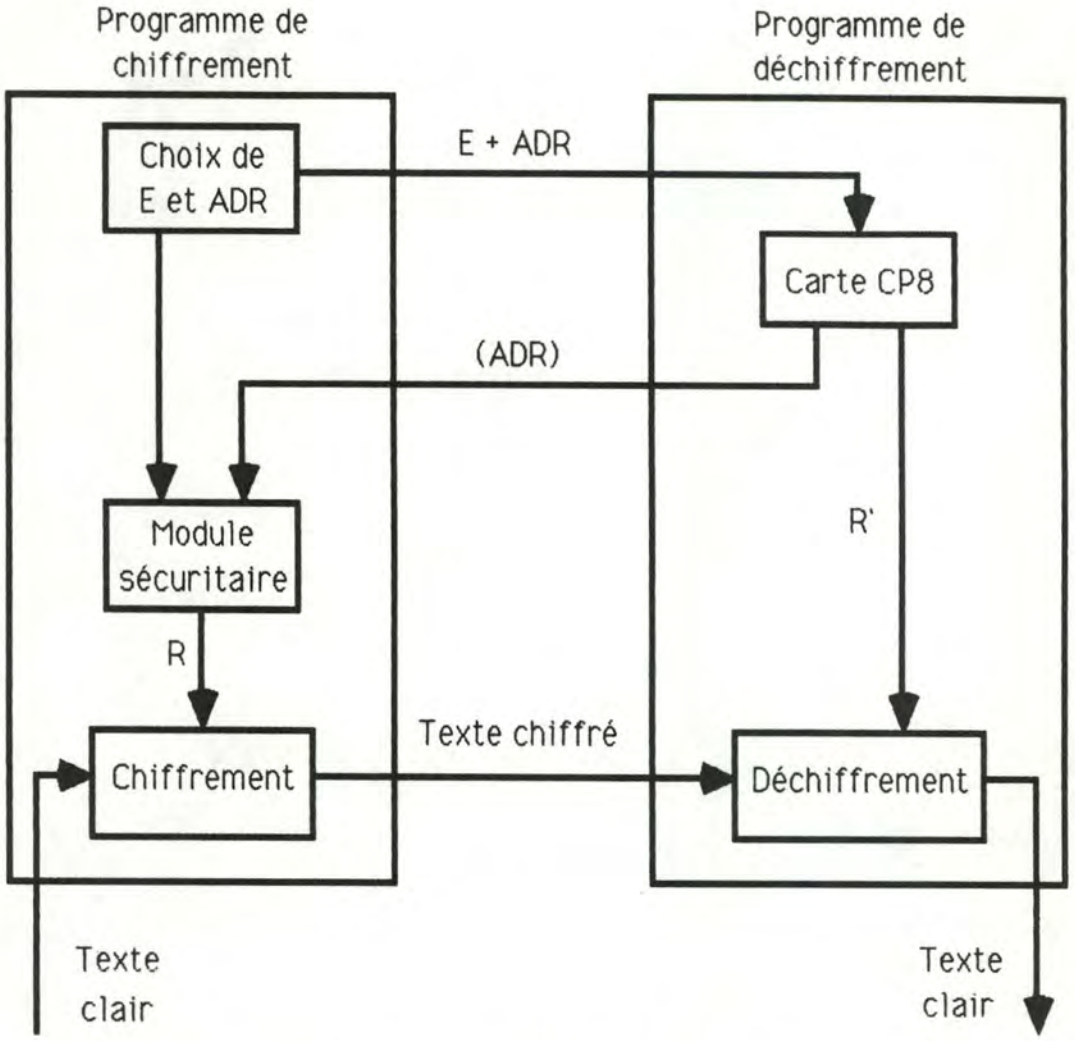


Fig 4.5 : Génération de clés de chiffrement

Remarque : la génération de clés de chiffrement serait également réalisable si le programme de chiffrement disposait de la carte CP8 et si le programme de déchiffrement disposait du module sécuritaire. Nous laissons au lecteur le soin d'imaginer les modifications que cela entraînerait dans le principe décrit ci-dessus.

4.1.2.2. Confidentialité des clés

Le caractère confidentiel des clés est assuré par :

- le chiffrement,
- l'inaccessibilité,
- la diversification,
- le blocage.

a) Chiffrement

Le chiffrement sert à éviter l'interception des clés pendant un dialogue. Chaque fois qu'une clé doit circuler de la carte vers un système externe (par exemple pour une authentification de carte) - ou l'inverse - elle est remplacée par un certificat. Ce dernier prouve au récepteur que l'émetteur connaît la clé, mais la connaissance du certificat ne permet pas à un éventuel fraudeur de deviner la clé utilisée.

L'authentification de l'utilisateur fait exception à cette règle. Cette exception se justifie aisément :

- L'utilisateur ne dispose pas de l'algorithme Télépass. Il lui est donc impossible de générer un certificat.

- L'authentification de l'utilisateur ne nécessite pas de dialogue avec des systèmes distants. Les possibilités d'interception de la clé secrète sont donc fortement réduites.

b) Inaccessibilité

Puisque les clés ne peuvent pas être interceptées pendant le dialogue, le seul moyen de les connaître est de remonter à leur source c'est-à-dire la carte, le module sécuritaire ou l'utilisateur.

- La carte

Toutes les clés sont stockées dans des mots qui ne sont accessibles de l'extérieur ni en lecture, ni en écriture. Dès qu'elles sont écrites dans la carte, seul le microprocesseur peut encore y avoir accès :

- en lecture : uniquement pour un usage interne.
- en écriture : uniquement lors du changement de la clé porteur.

- Le module sécuritaire

Le module sécuritaire est une carte CP8 utilisée uniquement pour l'implémentation des fonctions de sécurité et pour la diversification des clés (cfr point c. Diversification). Les clés qui y sont stockées sont donc protégées de la même manière que sur une carte CP8 classique.

- Les utilisateurs

Les utilisateurs (porteur et émetteur) doivent veiller eux-mêmes à ne pas divulguer leur clé, soit oralement, soit en la présentant à un appareil piraté.

De toute évidence, la sécurité des clés détenues par les utilisateurs n'est pas absolue. C'est pourquoi, certains masques prévoient des clés supplémentaires (par exemple, une forme digitalisée des empreintes digitales ou de la signature).

c) Diversification

Nous avons dit à plusieurs reprises que l'accès physique à la mémoire de la carte était impossible. C'est exact actuellement, mais il est tout à fait plausible que certaines protections physiques de la puce puissent être outrepassées dans un avenir plus ou moins proche (selon les capitaux investis dans ce type de recherches). Il est donc inconcevable de se fier aveuglément au principe d'inaccessibilité des clés.

La diversification constitue un système de sécurité complémentaire destiné à éviter que la découverte des clés d'une carte puisse servir pour une fraude à grande échelle. Cet objectif est réalisé en fournissant à chaque carte des clés qui lui sont propres.

Toutefois, un problème se pose si chaque carte dispose de clés propres. Comment un module sécuritaire - dont la capacité de mémorisation est limitée - peut-il connaître la clé interne de chacune des cartes avec lesquelles il est susceptible de dialoguer? De même, comment un émetteur peut-il connaître la clé émetteur des cartes qu'il distribue sans avoir recours à de longues listes de clés dont la confidentialité ne peut être que difficilement garantie?

Pour résoudre ce problème, le système de diversification permet de remplacer la mémorisation de clés par le calcul de clés. Le principe de la diversification (fig. 4.6.) est le suivant :

Lors de la personnalisation d'une carte - appelée couramment carte fille, les clés stockées dans la zone secrète sont choisies de manière à pouvoir être recalculées à tout moment.

A partir de la clé interne du module sécuritaire - également appelé carte mère, du numéro de série de la carte fille et d'un paramètre interne au module sécuritaire, l'émetteur calcule un certificat. Ce certificat constitue une clé de la carte fille.

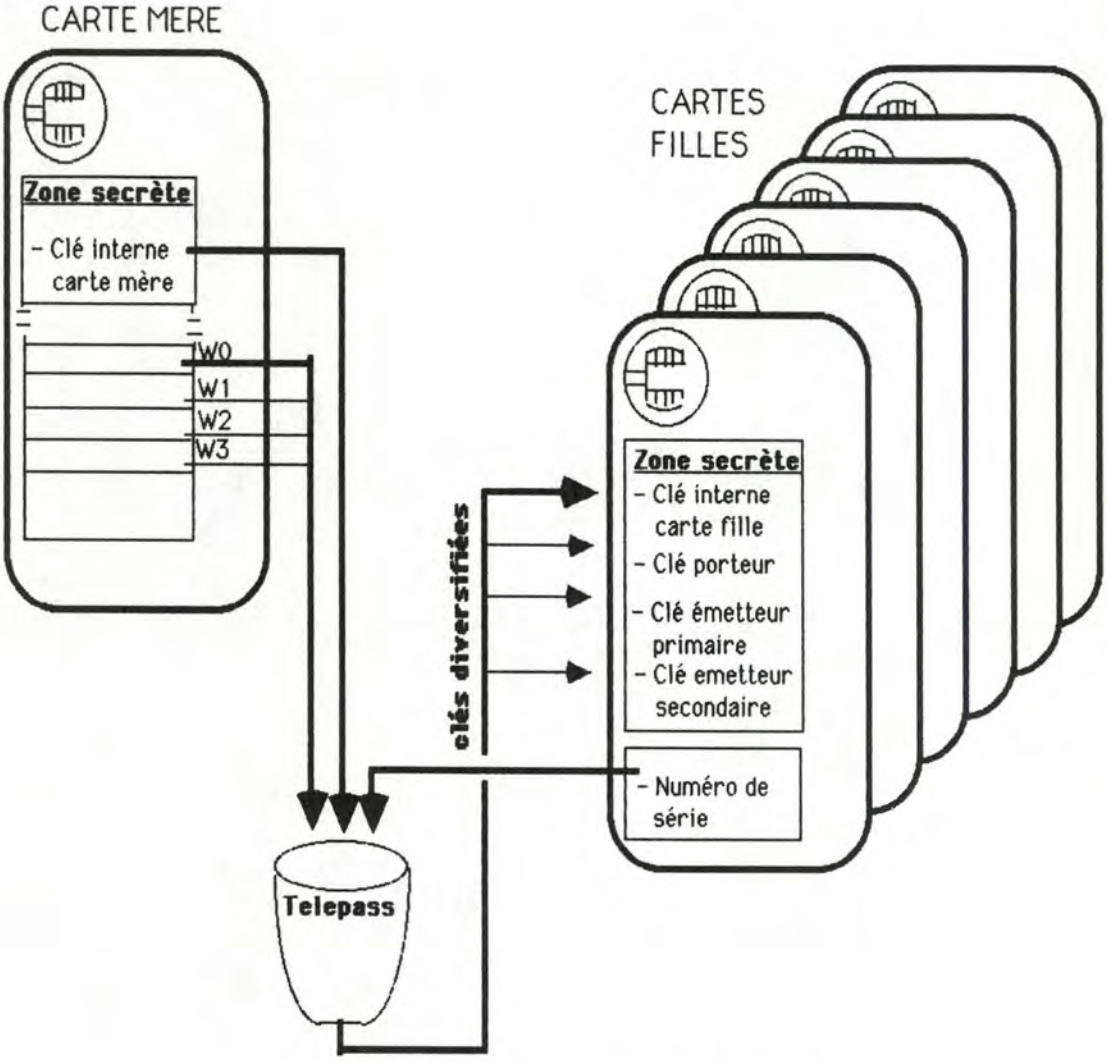


Fig. 4.6 : Principe de la diversification

Comme la figure 4.6 nous le montre, quatre des six clés de la zone secrète sont diversifiées au moyen d'une carte mère. Tout programme d'application disposant d'un module sécuritaire adéquat peut donc retrouver à tout moment la valeur de chacune de ces clés. Il lui suffit pour cela de lire le numéro de série de la carte avec laquelle il dialogue et, ensuite, de calculer un certificat en utilisant le paramètre interne correspondant à la clé qu'il veut connaître.

L'utilisation du numéro de série de la carte fille comme paramètre de diversification permet de diversifier les clés d'un grand nombre de cartes au départ d'un seul module sécuritaire.

La diversification ne sert pas uniquement pour le calcul des clés émetteur, porteur et interne. Elle sert également pour le calcul de la clé de fabrication. Dans ce cas, le principe de diversification est identique mais le module sécuritaire utilisé est appelé carte lot.

La figure 4.7 résume l'emploi de la diversification tout au long de la vie de la carte.

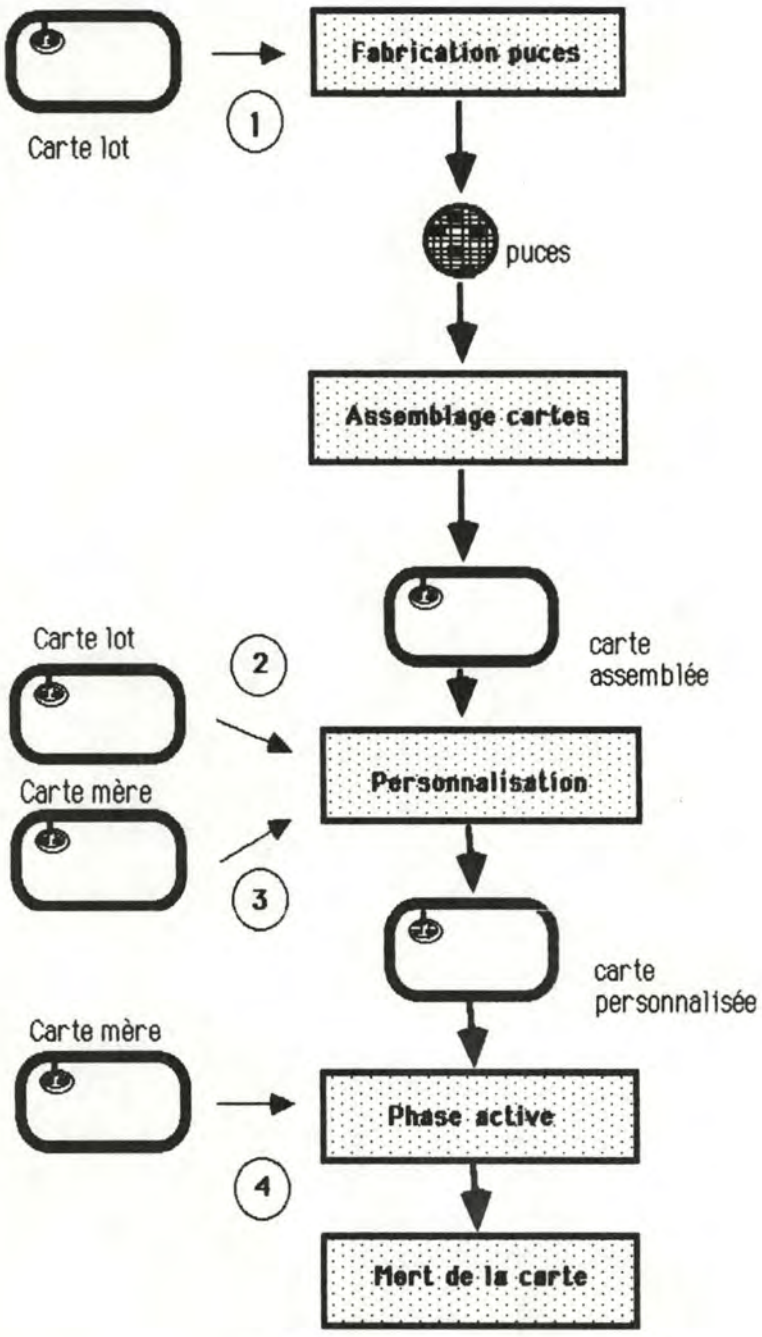


Fig. 4.7 : Diversification et vie de la carte

1. Le constructeur génère la clé de fabrication et écrit celle-ci dans la carte.

2. L'émetteur calcule la clé de fabrication et la présente à la carte afin de pouvoir réaliser les écritures requises pour la personnalisation.

3. L'émetteur génère les clés de la zone secrète (hormis la clé porteur de type deux) et les écrit dans la carte.

4. L'émetteur calcule la clé qui doit être utilisée dans une fonction de sécurité. Cette étape, contrairement aux trois premières qui n'ont lieu qu'une seule fois, a lieu chaque fois qu'une fonction de sécurité impliquant une clé diversifiée (autre que la clé de fabrication) doit être exécutée.

Comme nous le voyons sur la figure 4.7, les modules sécuritaires sont utilisés très fréquemment. En particulier, la carte mère est nécessaire pratiquement lors de chaque utilisation d'une carte CP8 par son porteur. Il est donc indispensable, dans le cadre d'applications décentralisées (systèmes off-line), de disposer de nombreuses copies de la carte mère.

C'est là que réside la faiblesse de la philosophie CP8. En effet, l'interception d'une seule de ces copies rendrait les fonctions de sécurité inopérantes contre le possesseur de cette copie. Le concepteur d'applications doit être bien conscient de cette possibilité afin d'être en mesure de prévoir des systèmes de sécurité complémentaires.

d) Blocage

Le blocage consiste à inhiber certaines instructions du microprocesseur s'il y a présomption de fraude. Il a pour but d'empêcher toute recherche systématique des clés d'authentification de l'utilisateur par essais et erreurs.

La carte s'autobloque lorsqu'elle s'aperçoit qu'une personne non habilitée essaie de l'utiliser. En d'autres termes, la carte se bloque lorsque l'utilisateur ne réussit pas à passer l'authentification avec succès. Le blocage est rendu possible grâce à la zone d'accès qui enregistre le résultat de chaque présentation de clé.

Remarques : - Il ne faut pas confondre le blocage et l'invalidation. Le blocage n'est pas définitif et est entièrement géré par la carte. Par contre, l'invalidation est définitive (cfr 3.4.1.5.

Mort de la carte) et peut avoir lieu soit sur ordre du programme d'application, soit sur un ordre du microprocesseur.

- Les présentations de clés de fabrication ne sont pas enregistrées dans la zone d'accès, mais dans le premier mot de la mémoire PROM. Le blocage survient lorsque trois clés de fabrication fausses ont été présentées successivement. Une carte ainsi bloquée ne peut pas être débloquée.

4.2. Polyvalence

Au niveau de la polyvalence, nous pouvons dire que la carte CP8 est un produit tout à fait satisfaisant. La grande polyvalence de la carte découle principalement de deux caractéristiques.

Tout d'abord, l'émetteur dispose d'une certaine liberté lors de la personnalisation de la carte, ce qui lui permet de configurer la carte en fonction de l'application. Plus précisément, il peut choisir :

- la taille respective des différentes zones (certaines zones peuvent même ne pas exister),

- le type de protection de la zone de transactions (cette zone peut être libre, protégée en lecture, protégée en écriture ou protégée en lecture et en écriture).

Par exemple, si la carte est utilisée comme matérialisation d'un droit d'accès, la zone de transactions est superflue. Par contre, si la carte sert de dossier portable, c'est la zone de transactions qui devient la plus importante et qui sera sans doute la plus grande.

La seconde raison pour laquelle la carte est polyvalente est qu'il n'existe pas de contraintes sémantiques sur le contenu de la zone confidentielle, de la zone de transactions et de la zone de lecture. Ces zones peuvent donc recevoir n'importe quel type d'informations.

4.3. Aspect multi-services

Des trois objectifs vus au chapitre trois, l'aspect multi-services est sans doute le seul dont la réalisation laisse à désirer. En effet, pour une carte multi-services, il serait souhaitable que :

- le nombre de services accessibles ne soit pas limité arbitrairement,

- chaque service puisse être totalement indépendant des autres.

Comme nous allons le voir, la carte CP8 ne possède pas tout à fait ces deux propriétés.

4.3.1. Nombre de services accessibles

Le masque décrit au chapitre trois a été conçu pour être utilisé au plus par deux émetteurs. Cela ne signifie pas pour autant que la carte ne permet la mise en oeuvre que de deux services simultanément. En effet, chaque émetteur est libre de subdiviser la mémoire de stockage qui lui a été attribuée, ce qui lui permet de proposer plusieurs services au porteur.

Le nombre de services accessibles au moyen de la carte n'est donc limité que par la taille de la mémoire disponible.

4.3.2. Indépendance des services

Le masque étudié présente deux caractéristiques qui garantissent une certaine indépendance aux services accessibles. Ce sont, d'une part, l'existence de deux clés émetteurs différentes et, d'autre part, la présence des bits C et CA parmi les bits systèmes (cfr point 3.2.1.2. Les mots).

Toutefois, le masque étudié présente aussi certaines faiblesses :

- Toute information accessible sur présentation de la clé émetteur primaire est aussi accessible sur présentation de la clé émetteur secondaire et vice versa.

- Lorsque la carte permet l'accès à plusieurs services, une partie de la mémoire est allouée a chaque service. Malheureusement, le masque ne gère pas le partage de la mémoire. Cela est particulièrement gênant pour la zone d'accès et pour la zone de transations :

- Puisque la zone d'accès est commune à tous les services, la présentation d'une clé émetteur fausse entraîne le blocage de toute la carte alors que seul le blocage du (des) service(s) concerné(s) par la clé présentée est souhaitable.

- Les protections de la zone de transactions s'étendent à toute la zone. Il ne peut donc pas exister simultanément des informations protégées et des informations non protégées dans la zone de transactions. Cela signifie que les protections en zone de transactions doivent être identiques pour tous les services accessibles via la carte.

- A priori, les programmes d'application sont autorisés à écrire n'importe où en zone de transactions. Il n'est donc pas impossible que des données relatives à un services soient écrites, par erreur ou par malveillance, dans une partie de la zone de transactions allouée à un autre service.

DEUXIEME PARTIE

REALISATION D'UN LOGICIEL D'AIDE A LA PROGRAMMATION D'APPLICATIONS

V. OBJECTIFS

La réalisation de ce logiciel a été guidée par quatre objectifs :

- la simplification du développement des programmes d'application,
- la portabilité du logiciel,
- la protection du logiciel,
- l'extensibilité du logiciel.

Dans ce chapitre, nous allons détailler ces quatre objectifs et présenter succintement les moyens mis en oeuvre pour les atteindre.

5.1. Simplification du développement des programmes d'application

Depuis la commercialisation de la carte CP8, il est apparu que la programmation d'applications basées sur la carte était complexe (nous en verrons les raisons au paragraphe 5.1.1). Il existe donc un besoin relativement important d'outils d'aide à la programmation d'applications. Dans ce contexte, la simplification du développement des programmes d'application constitue l'objectif principal du logiciel.

Pour atteindre cet objectif, la démarche adoptée a comporté deux étapes :

- l'identification des problèmes qui sont à l'origine de pertes de temps dans le développement,

- le choix d'une solution à ces problèmes.

5.1.1. Identification des problèmes

Les principales pertes de temps relevées au niveau du développement des programmes d'application découlent de trois problèmes :

- Le jeu d'instructions permettant la gestion de la carte n'est compatible avec aucun langage de programmation.

- Ces instructions sont de très bas niveau.

- Certaines fonctions de haut niveau qui se retrouvent dans la plupart des programmes d'application doivent être entièrement conçues et programmées lors du développement de chaque programme d'application.

5.1.1.1. Compatibilité des instructions

Comme nous l'avons vu dans la première partie, la carte à microprocesseur peut être considérée par le système externe abritant le programme d'application comme un périphérique particulier. Cependant, les systèmes externes concernés par ce logiciel (ordinateur central ou micro-ordinateur) ne perçoivent la carte à microprocesseur qu'au travers du ou des lecteurs-encodeurs auxquels ils sont connectés. Ces lecteurs-encodeurs disposent eux-mêmes d'un microprocesseur et d'un jeu d'instructions propre qui permet aussi bien la gestion du lecteur-encodeur que la gestion de la carte (1). Les programmes d'application n'utilisent donc pas les instructions du microprocesseur de la carte mais celles du microprocesseur du lecteur-encodeur.

Or, le format des instructions du lecteur-encodeur n'est compréhensible par aucun interpréteur ou compilateur. Pour développer un programme d'application, il faut donc créer une interface qui permette de gérer une carte à microprocesseur au départ du langage utilisé pour la programmation de l'application. Il faut souligner que la réalisation de cette interface nécessite une connaissance approfondie du protocole de communication avec un lecteur-encodeur et que cette connaissance fait défaut à la plupart des programmeurs.

(1) Le jeu d'instructions du lecteur-encodeur est fort semblable à celui de la carte. Nous ne le détaillons donc pas ici.

5.1.1.2. Niveau des instructions

Les instructions compréhensibles par le microprocesseur du lecteur-encodeur sont de très bas niveau, ce qui rend la gestion des données assez lourde. Pour nous en convaincre, imaginons l'exemple suivant :

Lorsque le porteur désire réaliser une transaction, le montant de la transaction est généralement écrit dans le(s) premier(s) mot(s) libre(s) de la zone de transactions.

Pour réaliser l'enregistrement du montant de la transaction, le programme d'application ne dispose que de l'instruction d'écriture qui doit recevoir comme paramètres :

- la zone d'écriture,
- l'adresse d'écriture dans la zone,
- la longueur des données,
- les données.

Pour écrire dans le premier mot libre de la zone de transactions, le programme d'application doit fournir l'adresse de ce mot à l'instruction d'écriture. Or, le programme d'application ne connaît pas cette adresse à priori. Il doit donc, pour la trouver, parcourir séquentiellement toute la zone de transactions jusqu'à ce qu'il rencontre un mot libre.

Supposons maintenant que le montant à écrire soit **1872**, ce qui se traduit en hexadécimal (sur quatre octets) par **00 00 07 1E**. Ces quatre octets, qui sont fournis comme paramètre à l'instruction d'écriture, ne seront pas écrits tels quels dans la carte. En effet, sur les 32 bits que comporte un mot, seuls 29 (ou 30) sont disponibles pour les données (cfr première partie, paragraphe 3.2.1.2.). En conséquence, après écriture et validation, le contenu du mot dans lequel le montant est enregistré est **80 00 07 1E** (2). Par la suite, lorsque le programme d'application sera amené à lire ce mot, il obtiendra la valeur **80 00 07 1E** et devra encore effectuer certaines opérations sur cette valeur pour en extraire la valeur réelle des données. Ces opérations s'avèrent particulièrement complexes lorsqu'une donnée peut chevaucher plusieurs mots.

Cet exemple montre que les instructions mises à la disposition du concepteur d'application présentent deux faiblesses :

- le seul mode d'accès aux données possible est l'accès direct,
- la seule structure de données manipulable est le mot tel qu'il est vu par le microprocesseur de la carte.

(2) La valeur 80 00 07 1E est fournie à titre exemplatif. Le contenu du mot dépend du code utilisé pour la représentation des données (ASCII, EBCDIC, base 2, ...) et du cadrage de ces données dans un mot.

5.1.1.3. Programmation des fonctions courantes

Certaines fonctions permettant la gestion d'une carte dépendent très peu du ou des services pour lesquels la carte est utilisée. Le déblocage de carte, le changement de clé porteur, la personnalisation, ... sont des exemples typiques de ce genre de fonctions.

Ces fonctions se retrouvent dans la plupart des programmes d'application. Le jeu d'instructions du microprocesseur ne comprend malheureusement pas d'instructions réalisant directement ces fonctions. Lors du développement de tout programme d'application, un temps considérable est donc perdu pour la définition et la programmation de ces fonctions.

5.1.2. Solutions envisagées

La première solution qui a été envisagée pour résoudre les problèmes exposés au paragraphe 5.1.1. consiste à fournir aux concepteurs de programmes d'application une librairie de primitives qui présentent les caractéristiques suivantes :

- Elles sont compatibles avec un langage de programmation évolué (le langage C) et constituent de ce fait une interface entre la carte et les programmes d'application (3).

- Elles sont beaucoup plus souples que les instructions compréhensibles par le microprocesseur du lecteur-encodeur au niveau de l'adressage et de la manipulation des données. Concrètement, elles permettent l'accès séquentiel et proposent une structure de données beaucoup plus riche que le mot.

- Elles couvrent des aspects de la gestion de la carte qui ne sont pas pris en compte par le jeu d'instructions du microprocesseur (par exemple le déblocage, le changement de clé porteur, ...).

Après analyse, il est apparu que la solution consistant à concevoir le logiciel comme une librairie de primitives pouvait être améliorée. En effet, certaines fonctions, telles que la personnalisation, présentent un caractère fortement interactif. Intégrer ces fonctions dans une librairie de primitives poserait des problèmes au niveau de la gestion d'écran.

En conséquence, une deuxième solution a été envisagée et finalement retenue. Cette solution consiste à prévoir deux modes d'utilisation différents pour le logiciel. Le logiciel peut être utilisé :

- comme une librairie de primitives directement utilisables au départ du programme d'application,

- comme un logiciel à part entière travaillant indépendamment de tout programme d'application.

Dans le premier cas (**mode librairie**), le logiciel doit être intégré au programme d'application au moment de la phase d'édition de liens. Les primitives de la librairie sont alors directement accessibles au programme d'application. Signalons que la fonction de personnalisation n'est pas accessible en mode librairie.

Dans le second cas (**mode interactif**), le logiciel se présente comme un programme interactif proposant des menus de fonctions à l'utilisateur. Le mode interactif est surtout utilisé pour les fonctions qui doivent gérer de nombreux paramètres (personnalisation, ...) et pour les fonctions simples qui ne participent pas directement à la gestion des données (déblocage, changement de clé porteur, ...), mais il peut aussi être utilisé pour travailler directement sur la zone de transactions.

(3) La décision de rendre les primitives compatibles avec le langage C a été dictée par deux raisons : d'une part, le langage C est fort répandu, et, d'autre part, il est prévu pour permettre des manipulations de données au niveau du bit.

5.2. Portabilité du logiciel

Puisque la carte à microprocesseur favorise l'exploitation de systèmes décentralisés, il est possible que des matériels de types différents soient utilisés pour une seule application. Il est donc indispensable que le logiciel soit adaptable à une vaste gamme d'ordinateurs (4).

Pour atteindre cet objectif, deux décisions ont été prises :

- Le logiciel doit être entièrement rédigé dans une version standardisée du langage C de manière à être compréhensible par la plupart des compilateurs C.

- Les dépendances vis-à-vis du matériel doivent être localisées dans une partie du logiciel de manière à n'avoir que cette partie à modifier pour adapter le logiciel à une nouvelle machine.

(4) Ces ordinateurs doivent obligatoirement être connectés à un ou plusieurs lecteurs-encodeurs de type TLP 124 PA8 et, éventuellement, à des lecteurs-encodeurs de type TLU 502.

5.3. Protection du logiciel

Le logiciel doit être protégé contre toute tentative de duplication ou d'utilisation non autorisée.

La protection du logiciel est assurée par deux cartes CP8 :

- La carte système qui doit être présentée avec sa clé porteur avant toute utilisation du logiciel.

- La carte d'habilitation qui doit être présentée avec sa clé porteur pour avoir accès à la fonction de personnalisation.

L'utilisation d'une deuxième carte pour protéger la fonction de personnalisation se justifie par l'importance que cette fonction revêt dans la lutte contre la fraude (cfr première partie, paragraphe 3.4.2.).

5.4. Extensibilité du logiciel

Tel qu'il est conçu actuellement, le logiciel ne couvre pas tous les aspects de la gestion d'une carte. En effet, certaines fonctions telles que la gestion de périphériques particuliers (certificateur, imprimante CP8, ...) ne sont pas implémentées. Il est donc encore possible de réduire le temps de développement des programmes d'application en ajoutant de nouvelles fonctionnalités au logiciel.

Le logiciel est conçu pour permettre une intégration aisée de ces nouvelles fonctionnalités. En effet, comme le logiciel est avant tout une librairie de primitives, il est relativement facile d'utiliser des primitives existantes pour en créer de nouvelles.

VI. PRINCIPES D'UTILISATION ET FONCTIONNALITES

Dans ce chapitre, nous allons décrire brièvement les principes d'utilisation du logiciel et les fonctionnalités qu'il propose. Nous procéderons de la manière suivante : pour chaque classe de fonctionnalités identifiée, nous présenterons d'abord, s'il y a lieu, quelques principes concernant le fonctionnement du logiciel, puis les fonctionnalités elles-mêmes.

Le logiciel est conçu pour permettre la personnalisation des cartes et pour faciliter leur exploitation pendant la phase active. Nous pouvons donc déjà relever deux classes de fonctionnalités : celles liées à la personnalisation et celles liées à l'exploitation. A ces deux classes, nous pouvons ajouter une troisième constituée des fonctionnalités liées à la protection du logiciel.

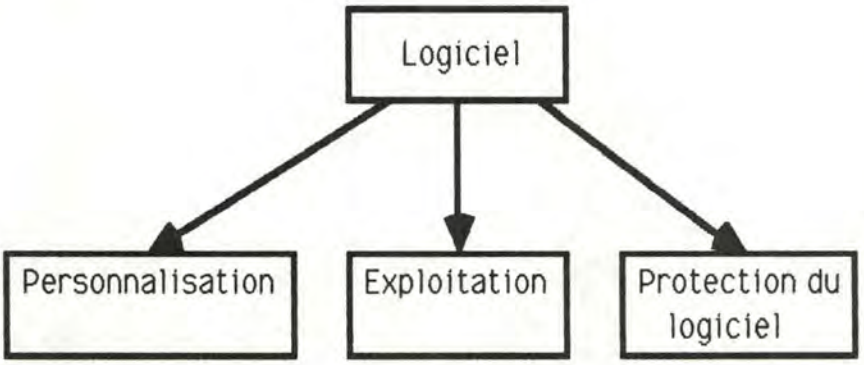


Fig. 6.1. : Fonctionnalités du logiciel

6.1. Personnalisation

6.1.1. Principes

Rappelons que la personnalisation consiste à inscrire dans la mémoire PROM de la carte les informations caractéristiques des services proposés ainsi que les données afférentes au destinataire final (le porteur).

Nous pouvons distinguer deux phases dans la personnalisation :

- la saisie (ou le calcul) des informations à écrire en mémoire,
- l'écriture de ces informations dans la mémoire PROM.

Selon que l'on considère ces deux phases comme un tout ou comme deux opérations distinctes, nous pouvons aboutir à deux modes de personnalisation différents :

- la personnalisation à la carte qui regroupe les opérations de saisie et d'écriture,
- la personnalisation en grande série qui différencie les deux opérations.

Dans le premier cas, la personnalisation est entièrement interactive et exécutée en une seule fois, carte après carte.

Dans le second cas, la personnalisation se fait par lot de cartes. La personnalisation d'un lot de cartes comporte d'abord la saisie interactive des informations pour toutes les cartes du lot, puis, l'écriture automatique de ces informations dans la mémoire des cartes.

Dans les deux cas, nous constatons que, dans le cadre d'une application, les cartes sont souvent configurées de la même manière et que seules les informations liées au porteur ou au numéro de série de la carte varient d'une carte à l'autre. D'où l'idée de définir deux étapes dans la saisie des paramètres de personnalisation :

- Dans la première étape, le logiciel va saisir les paramètres invariants qui vont servir à définir la configuration de la carte dans le cadre d'une application, c'est-à-dire, la taille des zones, la taille des clés, ... Une configuration ne sera définie qu'une fois pour tout un lot de cartes.

- Dans la seconde étape, le logiciel va saisir les paramètres directement liés au porteur ou au numéro de série de la carte, c'est-à-dire, le contenu de la zone confidentielle, le contenu de la zone de lecture et, éventuellement, la valeur de certaines clés secrètes. Cette étape sera réalisée pour chaque carte du lot.

L'avantage de cette découpe est de ne pas devoir demander à l'utilisateur de fournir tous les paramètres de configuration pour chaque carte à personnaliser; et cela, quel que soit le mode de personnalisation adopté.

La figure 6.2 résume le fonctionnement de la personnalisation.

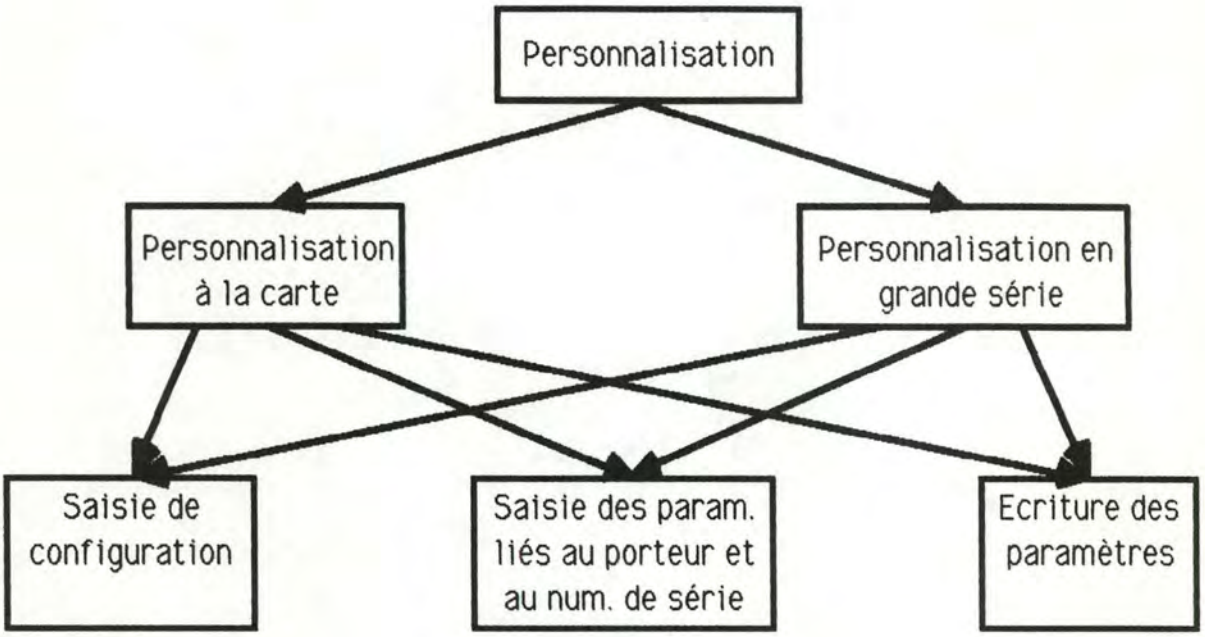


Fig. 6.2. : Fonctionnement de la personnalisation

Remarque : les deux modes de personnalisation (à la carte ou en grande série) sont prévus dans le logiciel mais ne sont accessibles que via le mode interactif.

6.1.2. Fonctionnalités

Au niveau fonctionnel, la personnalisation peut être considérée comme une opération élémentaire. Elle constitue donc la seule fonctionnalité de la classe.

6.2. Exploitation pendant la phase active

Les fonctionnalités liées à l'exploitation d'une carte pendant sa phase active couvrent quatre domaines :

- la gestion de la configuration de la zone de transactions,
- la gestion des données,
- la sécurité de l'application,
- les changements d'état de la carte.

6.2.1. Gestion de la configuration de la zone de transactions

6.2.1.1. Principes

Nous avons vu dans la première partie que, si un émetteur désire proposer plusieurs services au porteur, il doit allouer une partie de la mémoire dont il dispose à chaque service. Dans cette optique, le logiciel prévoit la possibilité de subdiviser la zone de transactions en un nombre quelconque de zones appelées zones de service.

6.2.1.2. Fonctionnalités

La création d'une zone de service à n'importe quel moment pendant la phase active constitue la seule fonctionnalité de la classe.

6.2.2. Gestion des données

6.2.2.1. Principes

Nous avons signalé au chapitre précédent que le logiciel est plus riche que le jeu d'instructions du microprocesseur du lecteur-encodeur au niveau de la gestion des données et, plus particulièrement, au niveau des modes d'accès et au niveau des structures de données disponibles.

Le logiciel propose trois modes d'accès distincts :

- L'accès direct par adressage absolu : le programme d'application doit fournir l'adresse de la donnée par rapport au début de la mémoire PROM. Ce mode d'accès correspond à celui utilisé par les instructions du microprocesseur de la carte.

- L'accès direct avec adressage relatif : le programme d'application doit fournir l'adresse de la donnée par rapport au début de la zone où doit avoir lieu l'accès. Dans ce cas, la première donnée de la zone se trouve à l'adresse 1. Ce mode d'accès correspond à celui utilisé par les instructions du microprocesseur du lecteur-encodeur.

- L'accès séquentiel.

Remarque : dans le cas de l'accès direct, toutes les adresses sont exprimées en mots.

Au niveau des structures de données, le logiciel propose deux possibilités :

- Le mot : un mot est un ensemble de 32 bits (comprenant des bits systèmes) tel qu'il est manipulé par le microprocesseur de la carte. Normalement, cette structure de données devrait être très peu utilisée en raison de la lourdeur de sa gestion.

- L'enregistrement : un enregistrement est un ensemble comprenant un ou plusieurs champs, parfois de types différents (numérique, caractère, ...), regroupés sous un seul nom. L'enregistrement correspond à la structure en C ou au record en Pascal.

6.2.2.2. Fonctionnalités

Au niveau de la gestion des données, le logiciel propose des fonctions couvrant de nombreux aspects de la gestion des enregistrements et des mots, c'est-à-dire la lecture et l'écriture sur carte, la saisie et l'affichage à l'écran.

6.2.3. Sécurité de l'application

6.2.3.1. Principes

Rappelons que la sécurité d'une application repose principalement sur cinq fonctions de sécurité : l'authentification de l'utilisateur, l'authentification de la carte, la certification, la génération de clés de chiffrement et la signature électronique.

6.2.3.2. Fonctionnalités

L'authentification de l'utilisateur, l'authentification de la carte et la certification sont implémentées dans le logiciel; les deux autres fonctions, étant d'usage moins courant, ne le sont pas. Cependant, le logiciel permet de faciliter l'implémentation de ces fonctions en proposant des primitives de génération de certificat et de diversification de clés.

Remarque : dans une prochaine version du logiciel, il est vraisemblable que les cinq fonctions de sécurité seront implémentées et directement accessibles au départ des programmes d'application.

6.2.4. Changements d'état de la carte

6.2.4.1. Principes

Au cours de la phase active, le programme d'application peut provoquer trois types de changements d'état pour une carte :

- le déblocage,
- l'invalidation,
- le changement de clé porteur.

6.2.4.2. Fonctionnalités

Les opérations de déblocage, d'invalidation et de changement de clé porteur constituent chacune une fonctionnalité du logiciel.

La figure 2.3 reprend les fonctionnalités du logiciel liées à l'exploitation de la carte pendant la phase active.

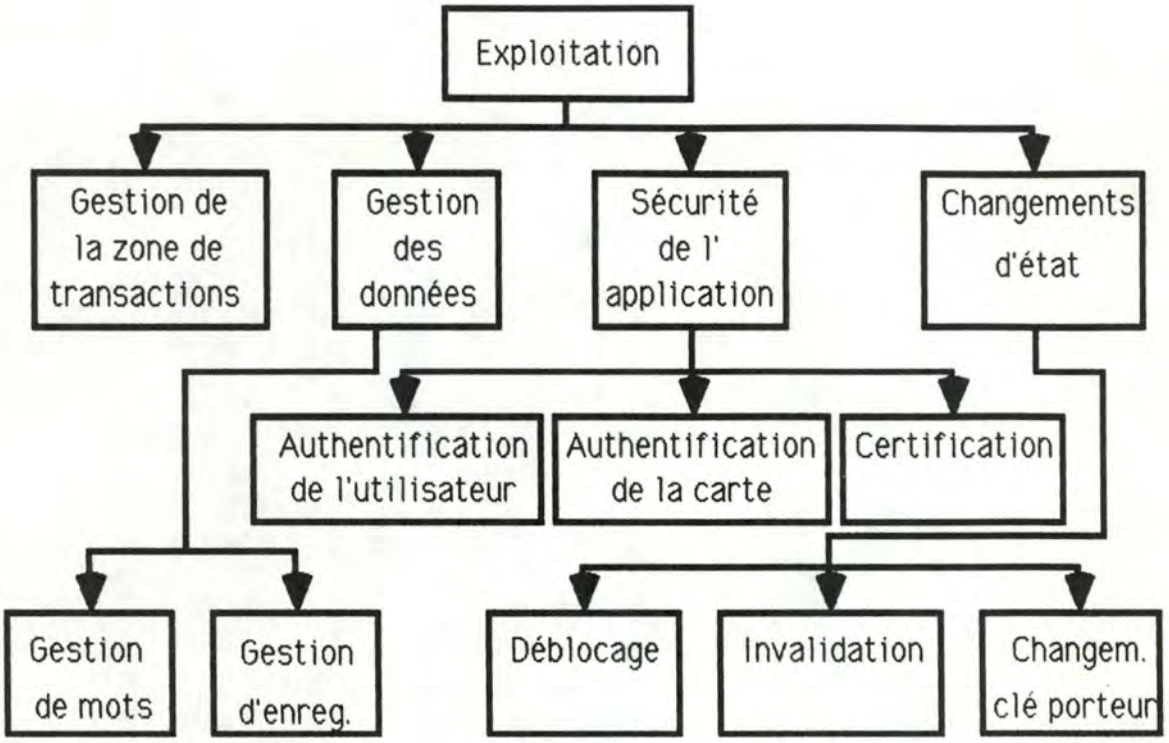


Fig 6.3. : Fonctionnalités liées à l'exploitation

Remarques : - Toutes les fonctionnalités décrites au paragraphe 6.2. sont accessibles aussi bien en mode interactif qu'en mode librairie.

- Comme la personnalisation n'est qu'un cas particulier d'exploitation de la carte, elle est aussi basée sur les fonctionnalités décrites aux paragraphes 6.2.2. et 6.2.3. Cependant, l'utilisateur n'a pas conscience de l'existence de ces fonctionnalités lorsqu'il personnalise une carte.

6.3. Protection du logiciel

Comme nous l'avons vu au chapitre précédent, le logiciel est protégé par deux cartes CP8 (la carte système et la carte d'habilitation) contre toute tentative de duplication ou d'utilisation non autorisée.

Pour mettre en oeuvre cette protection, le logiciel dispose de primitives qui lui permettent de :

- contrôler l'authenticité des cartes qui lui sont présentées,
- vérifier l'exactitude des clés présentées.

VII. ARCHITECTURE LOGIQUE

7.1. Présentation de la hiérarchie

L'architecture logique du logiciel est structurée hiérarchiquement et composée de modules reliés entre eux par une relation de type **utilise**.

Une relation de type **utilise** entre deux modules A et B exprime que le fonctionnement correct du module A dépend de la disponibilité d'une version correcte du module B. L'emploi d'une telle relation permet de définir des niveaux tels que :

Au niveau 1, se trouvent les modules A pour lesquels il n'existe pas de modules B tels que B utilise A.

Au niveau i, se trouvent les modules A pour lesquels :

- Il existe un ou plusieurs modules B de niveau i-1 tels que B utilise A.
- S'il existe un module C tel que A utilise C, alors C est de niveau supérieur à i.

La figure 7.1 schématise la place respective des différents niveaux.

Fig 7.1. : Emplacement respectif des niveaux

En première approximation, nous pouvons dire qu'un module se situera à un niveau d'autant plus élevé dans la hiérarchie (c'est-à-dire d'autant plus proche du niveau 1) que la tâche qu'il doit accomplir est proche d'une fonctionnalité identifiée au chapitre six. C'est ainsi que les modules de personnalisation, de déblocage, ... se retrouvent au niveau 1, tandis que les modules fortement liés à l'environnement matériel (interface avec la carte, ...) se retrouvent aux niveaux 3 et 4.

La coordination entre les différents modules de la hiérarchie est assurée, selon le mode de fonctionnement du logiciel, par le programme d'application (mode librairie) ou par un programme coordinateur inclus dans le logiciel (mode interactif).

La figure 7.2 représente l'architecture logique du logiciel.

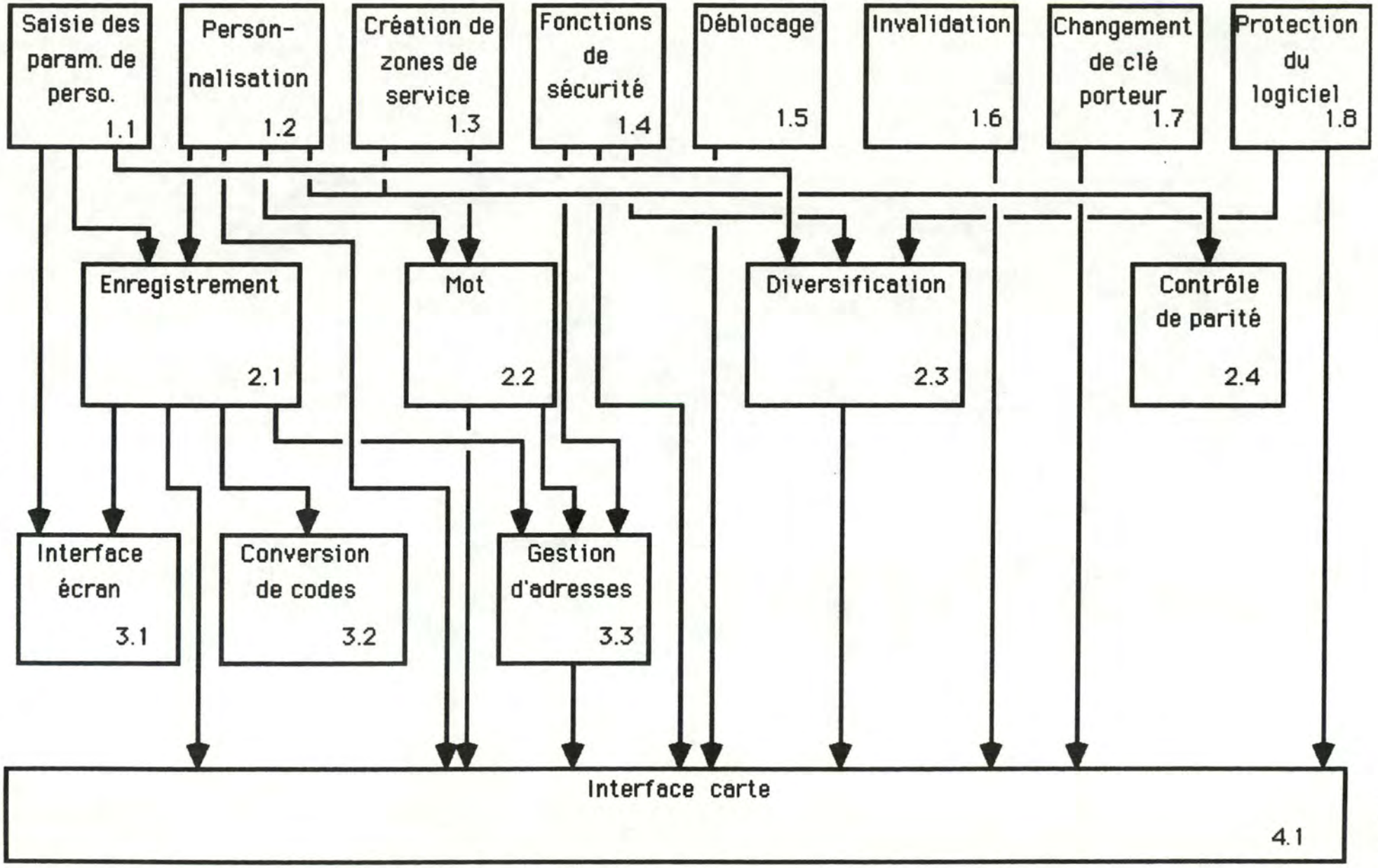


Fig 7.2 : Architecture logique

7.2. Description sommaire des modules

7.2.1. Module 1.1 : Saisie des paramètres de personnalisation

Le module 1.1 réalise la saisie interactive des paramètres de personnalisation (cfr paragraphe 6.1.1) et leur mise en place dans deux structures de données destinées à être exportées vers le module de personnalisation.

Il est constitué de deux composants : le composant de saisie de configuration et celui de saisie des paramètres liés au porteur et au numéro de série de la carte. Chacun des composants produit une des structures de données qui seront exportées vers le module de personnalisation.

7.2.2. Module 1.2 : Personnalisation

Le module 1.2 réalise la personnalisation d'une carte au départ des structures de données qui lui sont fournies par le module de saisie des paramètres de personnalisation.

7.2.3. Module 1.3 : Création de zones de service

Le module 1.3 réalise la création d'une zone de service de taille donnée (cfr paragraphe 6.2.1.). Cette zone de service doit se trouver entièrement en zone de transactions et ne peut chevaucher aucune autre zone de service.

7.2.4. Module 1.4 : Fonctions de sécurité

Le module 1.4 est chargé d'assurer la sécurité de l'application.

Il est constitué de trois composants qui sont chacun attachés à la réalisation d'une fonction de sécurité. Les trois fonctions de sécurité concernées par ces composants sont l'authentification de l'utilisateur, l'authentification de la carte et la certification.

7.2.5. Module 1.5 : Déblocage

Le module 1.5 réalise le déblocage d'une carte, c'est-à-dire la présentation à la carte de la clé porteur et de la clé émetteur primaire.

7.2.6. Module 1.6 : Invalidation

Le module 1.6 réalise l'invalidation d'une carte.

7.2.7. Module 1.7 : Changement de clé porteur

Le module 1.7 permet le changement de la clé porteur d'une carte.

7.2.8. Module 1.8 : Protection du logiciel

Le module 1.8 contrôle l'authenticité de la carte système et de la carte d'habilitation et vérifie l'exactitude des clés porteurs associées à ces cartes.

Ce module doit être activé avant toute utilisation du logiciel en mode interactif ou en mode librairie.

7.2.9. Module 2.1 : Enregistrement

L'enregistrement est un type abstrait de données. Ce type abstrait permet de représenter et de manipuler des structures de données complexes (un enregistrement équivaut approximativement à une structure en C).

7.2.10. Module 2.2 : Mot

Le module 2.2 réalise la lecture et l'écriture d'un mot dans la mémoire de stockage de la carte.

A ce niveau, les trois modes d'accès prévus dans le logiciel (accès séquentiel, accès direct par adressage relatif et accès direct par adressage absolu) sont utilisables.

7.2.11. Module 2.3 : Diversification

Le module 2.3 réalise la diversification de la clé de fabrication, des clés émetteurs primaire et secondaire, de la clé interne, et de la clé porteur de type 1.

Une clé diversifiée par ce module a une taille fixe de 8 octets et est calculée a partir d'un paramètre interne prédéterminé. Si le concepteur d'application désire des clés de taille différente, ou calculées à partir de paramètres internes différents, il devra redéfinir sa propre fonction de diversification.

7.2.12. Module 2.4 : Contrôle de parité

Le module 2.4 réalise un contrôle de parité sur des données.

Le contrôle de parité sert à garantir qu'une information stockée dans la carte a été écrite correctement. Il est en effet possible que, suite à une erreur de manipulation (réécriture sur un mot non validé) ou à un léger défaut de fabrication, l'information réellement écrite ne corresponde pas à celle que le programme d'application désirait écrire. L'adjonction de bits de parité à l'information écrite permet, lors d'une lecture, de vérifier la bonne écriture de l'information.

Attention, l'utilisation de bits de parité diminue le nombre de bits utiles dans un mot !

7.2.13. Module 3.1 : Interface écran

Le module 3.1 réalise la saisie à l'écran des données dont le type est spécifique au logiciel (ces types seront vus au paragraphe 7.3.9.1). Il réalise tous les contrôles syntaxiques et certains contrôles sémantiques.

7.2.14. Module 3.2 : Conversion de codes

Le module 3.2 transforme la représentation des données : il exprime dans un code cible les données initialement exprimées dans un code source.

Ce module constitue un outil destiné à minimiser l'encombrement des données avant leur stockage sur une carte CP8.

7.2.15. Module 3.3 : Gestion des adresses

Le module 3.3 est dédicacé à la gestion des adresses. Il prend en charge la conversion d'adresses (adresses absolues en adresses relatives et vice versa), la recherche de mots libres, ...

7.2.16. Module 4.1 : Interface carte

Le module 4.1 permet aux autres modules de dialoguer avec le lecteur-encodeur et, de là, avec la carte. Le module d'interface carte constitue approximativement une réplique compréhensible par le langage C du jeu d'instructions du microprocesseur du lecteur-encodeur.

Ce module a pour principale fonction de cacher au reste du logiciel le protocole de communication avec le lecteur-encodeur et la carte.

7.3. Spécification des modules

Tous les modules de la hiérarchie sont des modules de traitement, à l'exception du module enregistrement qui est un module de données (type abstrait).

La spécification d'un module de traitement comporte cinq parties :

- une description rapide de la fonction du module,
- la description des arguments du module,
- la spécification des pré-conditions que les arguments doivent vérifier pour que le module produise le résultat attendu,
- la description des résultats du module,
- la spécification des post-conditions que les résultats vérifient après l'exécution du module.

La spécification d'un type abstrait de données comporte deux parties :

- la description détaillée du type abstrait,
- la spécification des opérations réalisables sur ce type abstrait (la spécification d'une opération est identique à la spécification d'un module de traitement).

Remarques : - Un module (ou un composant) ne correspond pas forcément à une primitive en langage C. Un module ne constitue donc pas une unité d'exécution pour une machine réelle, mais pour une machine abstraite.

- Lors de la première lecture de ce paragraphe, il est conseillé de commencer par lire le paragraphe 7.3.9 (Enregistrement) car la notion d'enregistrement doit être comprise pour aborder ce qui suit.

- L'architecture physique et les algorithmes abstraits ne sont pas définis dans ce mémoire. Dès lors, afin de faciliter l'implémentation du logiciel, les spécifications de certains modules sont accompagnées de recommandations de réalisation.

7.3.1. Module 1.1 : Saisie des paramètres de personnalisation

7.3.1.1. Saisie de configuration

Fonction : saisir les paramètres permettant de définir la configuration d'un lot de cartes.

Arg : -

Pré : -

Rés : CONFIG. CONFIG est une structure de données comprenant :

- le nom de la configuration,
- l'adresse de début de la clé émetteur secondaire,
- la taille de la clé émetteur secondaire,
- la taille de la clé interne,
- la taille de la clé porteur de type 1,
- une variable signalant les clés qui doivent être diversifiées (les clés concernées par cette variable sont les deux clés émetteurs, la clé interne et la clé porteur de type 1),
- la taille de la zone d'accès,
- la taille de la zone confidentielle,
- la taille de la zone de lecture,
- la protection de la zone de transactions en lecture,
- la protection de la zone de transactions en écriture,
- la référence à un type d'enregistrement définissant la configuration de la zone confidentielle,
- la référence à un type d'enregistrement définissant la configuration de la zone de lecture.

La zone confidentielle et la zone de lecture sont chacune considérées comme un enregistrement particulier, c'est-à-dire comme une structure de données directement manipulable par le logiciel. Généralement, lors de l'activation du composant de saisie de configuration, il y a création en mémoire centrale de deux types d'enregistrements qui décrivent respectivement le schéma des deux zones. Précisons que, lorsque l'une des deux zones n'existe pas, le type d'enregistrement qui y correspond n'est pas créé.

Post : Les deux types d'enregistrements référencés dans CONFIG sont accessibles à la fin de l'exécution du composant.

La taille de la zone confidentielle est égale à la longueur de tous les champs de l'enregistrement qui décrit le schéma de la zone. Il en va de même pour la zone de lecture.

Une clé de taille nulle ne doit pas être diversifiée.

L'adresse de la clé émetteur secondaire est une adresse située dans l'espace d'adressage de la carte et telle que la clé émetteur

secondaire ne chevauche ni la clé de fabrication, ni la zone de fabrication.

La somme des tailles qui figurent dans CONFIG ne peut dépasser la taille de la mémoire disponible sur la carte, c'est-à-dire l'espace compris entre le début de la clé émetteur secondaire et le début de la zone de fabrication.

Si le nom de la configuration est vide, cela correspond à une demande d'annulation de la saisie de configuration de la part de l'utilisateur.

7.3.1.2. Saisie des paramètres liés au porteur ou au numéro de série

Fonction : saisir ou calculer les paramètres de personnalisation liés au porteur ou au numéro de série de la carte à personnaliser.

Arg : CONFIG.

Pré : CONFIG est une structure de données qui vérifie les post-conditions définies au paragraphe 7.3.1.1.

Les types d'enregistrements référencés dans CONFIG sont en mémoire centrale.

Rés : PARAM. PARAM est une structure de données comprenant :

- la référence de l'enregistrement contenant la valeur de toutes les clés secrètes, hormis la clé de fabrication et la clé porteur de type 2,
- la référence de l'enregistrement contenant toutes les informations devant être stockées en zone confidentielle,
- la référence de l'enregistrement contenant toutes les informations devant être stockées en zone de lecture.

Post : Les trois enregistrements référencés dans PARAM doivent, s'ils existent, être en mémoire centrale à la fin de l'exécution du composant.

Les deux derniers enregistrements référencés dans PARAM sont des occurrences des types d'enregistrements référencés dans CONFIG.

Remarque : Les clés secrètes seront diversifiées ou saisies à l'écran en fonction de ce qui est spécifié dans CONFIG.

7.3.2. Module 1.2 : Personnalisation

Fonction : personnaliser une carte.

Arg : CONFIG,
PARAM.

Pré : CONFIG est une structure de données qui vérifie les post-conditions définies au paragraphe 7.3.1.1.

PARAM est une structure de données qui vérifie les post-conditions définies au paragraphe 7.3.1.2.

Les trois enregistrements référencés dans PARAM doivent, s'ils existent, être en mémoire centrale.

Rés : RC.

Post : RC a pour valeur :
0 : La personnalisation s'est déroulée correctement.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : La carte est bloquée.
3 : La carte est invalide.

7.3.3. Module 1.3 : Création de zones de service

Fonction : créer une zone de service de référence REF et de longueur LGR à l'adresse ADR (ADR est une adresse relative par rapport au début de la zone de transactions).

Arg : ADR,
LGR,
REF.

Pré : -

Rés : RC.

Post : RC a pour valeur :
0 : La zone de service est créée.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : ADR ne fait pas référence à un mot situé en zone de transactions.
3 : ADR fait référence à un mot situé dans une zone de service.

4 : ADR fait référence à un mot non libre situé dans la zone de transactions.

5 : Il ne reste pas suffisamment de place pour réserver toute la zone de service.

6 : La carte est bloquée.

7 : La carte est invalide.

Recommandations de réalisation

Les zones de service ne peuvent être gérées de la même manière que les sept zones prévues par le masque. En effet, contrairement à ces sept zones, les zones de service ne sont pas définies lors de la personnalisation, mais à n'importe quel moment pendant la phase active de la carte. Leurs caractéristiques (taille, emplacement et référence) ne peuvent pas être spécifiées dans la zone de fabrication puisque l'écriture en zone de fabrication est impossible pendant la phase active. Pour spécifier les caractéristiques des zones de service, il est recommandé d'avoir recours à des headers.

Un header (fig. 7.3) est un mot particulier qui est placé au début d'une zone de service et qui en spécifie la référence et la taille.

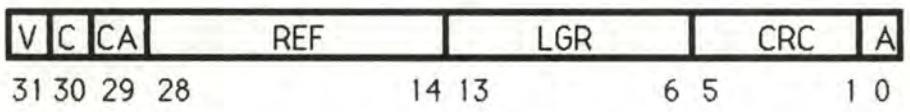


Fig7.3 : Header de zone de service

- V, C et CA: sont les bits systèmes vus au paragraphe 3.2.1.2. de la première partie.

- REF : est une référence qui permet de distinguer les zones de service entre elles. REF est une zone de 15 bits.

- LGR : est une zone de 8 bits qui spécifie la longueur de la zone de service. Cette longueur est exprimée en mots.

- CRC : est une zone de 5 bits qui contient les bits de parité relatifs aux 27 autres bits du mot.

- A : est le bit applicatif. Il permet de distinguer un header d'un mot utilisé pour stocker des données propres à un service.

7.3.4. Module 1.4 : Fonctions de sécurité

7.3.4.1. Authentification de l'utilisateur

L'authentification de l'utilisateur se subdivise en trois sous-composants : l'authentification du porteur, l'authentification de l'émetteur primaire et l'authentification de l'émetteur secondaire.

a) Authentification du porteur

Fonction : authentifier le porteur.

Arg : CLE-P : clé secrète du porteur.

Pré : -

Rés : RC.

Post : RC a pour valeur :
0 : La clé présentée est correcte.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : La zone d'accès a enregistré la présentation d'une clé porteur fausse.
3 : La zone d'accès a enregistré la présentation de deux clés porteurs fausses.
4 : La carte est bloquée.
5 : La zone d'accès est saturée.

b) Authentification de l'émetteur primaire

Fonction : authentifier l'émetteur primaire.

Arg : CLE-EP : clé émetteur primaire.

Pré : -

Rés : RC.

Post : RC a pour valeur :
0 : La clé présentée est correcte.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : La carte est bloquée.
3 : La zone d'accès est saturée.

c) Authentification de l'émetteur secondaire

Fonction : authentifier l'émetteur secondaire.

Arg : CLE-ES : clé émetteur secondaire.

Pré : -

Rés : RC.

Post : RC a pour valeur :
0 : La clé présentée est correcte.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : La carte est bloquée.
3 : La zone d'accès est saturée.

7.3.4.2. Authentification de la carte

Fonction : authentifier la carte.

Arg : -

Pré : -

Rés : RC.

Post : RC a pour valeur :
0 : La carte est authentique.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : La carte n'est pas authentique.

7.3.4.3. Certification

Fonction : générer un certificat relatif au contenu du mot situé à l'adresse ADR dans la zone ZONE.

Arg : E : message d'entrée,
ADR,
ZONE.

Pré : ZONE fait référence à une zone existante.

Rés : R,
RC.

Post : RC a pour valeur :
0 : R est le certificat obtenu en activant Télépass avec comme paramètres la clé interne de la carte, l'adresse ADR, le contenu du mot dont l'adresse est ADR et le message d'entrée E.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : Les conditions de lecture ne sont pas remplies.
3 : La carte est bloquée.
4 : ZONE fait référence à une zone inconnue.

Remarque : les zones connues à ce niveau sont : la zone d'accès, la zone confidentielle, la zone de lecture, la zone de transactions, les zones de services et la zone de fabrication.

7.3.5. Module 1.5 : Déblocage

Fonction : débloquer une carte.

Arg : CLE-P : clé porteur,
CLE-EP : clé émetteur primaire.

Pré : -

Rés : RC.

Post : RC a pour valeur :
0 : La carte est débloquée.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : La clé porteur présentée est fausse.
3 : La clé émetteur primaire présentée est fausse.
4 : Les deux clés présentées sont fausses.
5 : La zone d'accès est saturée.

7.3.6. Module 1.6 : Invalidation

Fonction : invalider une carte.

Arg : -

Pré : -

Rés : RC.

Post : RC a pour valeur :
0 : La carte est invalidée.
1 : Il n'y a pas de carte dans le lecteur-encodeur.

7.3.7. Module 1.7 : Changement de clé porteur

Fonction : remplacer la clé porteur de type 1 d'une carte par la clé porteur de type 2.

Arg : CLE-P1 : clé porteur de type 1,
CLE-P2 : clé porteur de type 2.

Pré : -

Rés : RC.

Post : RC a pour valeur :
0 : Le changement s'est déroulé correctement.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : La clé porteur de type 1 présentée est fausse.
3 : La carte est bloquée.
4 : La carte est invalide.

7.3.8. Module 1.8 : Protection du logiciel

7.3.8.1. Protection par carte système

Fonction : authentifier la carte système et vérifier l'exactitude de la clé porteur présentée.

Arg : CLE-P : clé porteur de la carte système,

Pré : -

Rés : RC.

Post : RC a pour valeur :
0 : La carte est authentique et la clé présentée est correcte.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : La carte n'est pas authentique.
3 : La carte est authentique et la clé présentée est fausse.
4 : La carte est authentique mais bloquée.

7.3.8.2. Protection par carte d'habilitation

Fonction : authentifier la carte d'habilitation et vérifier l'exactitude de la clé porteur présentée.

Arg : CLE-P : clé porteur de la carte d'habilitation,

Pré : -

Rés : RC.

Post : RC a pour valeur :
0 : La carte est authentique et la clé présentée est correcte.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : La carte n'est pas authentique.
3 : La carte est authentique et la clé présentée est fausse.
4 : La carte est authentique mais bloquée.

Recommandations de réalisation

Pour authentifier la carte système et la carte d'habilitation, le logiciel doit contrôler leur clé interne et donc, connaître la valeur de ces clés. Par conséquent, ces clés ne doivent pas être diversifiées (sinon la production en grande série du logiciel devient impossible).

Pour vérifier l'exactitude de la clé porteur associée à chaque carte, il suffit de présenter la clé à la carte et de tester la valeur du code RC renvoyé par la primitive de présentation de clé.

7.3.9. Module 2.1 : Enregistrement

7.3.9.1. Description de l'enregistrement

L'enregistrement est un ensemble de champs regroupés sous un seul nom. Il peut être représenté de la manière suivante :

```
Nom d'enregistrement {
        Nom de champ : type du champ,
                .
                .
                .
        Nom de champ : type du champ
        }
```

Tout comme en C, il y a lieu de distinguer type d'enregistrement et occurrence d'enregistrement. Dans l'exemple qui suit, **date** constitue un type d'enregistrement et **d1** est une occurrence de **date**.

```
date {                          d1 {
     jour  : int,                    31,
     mois  : int,                    8,
     année : int                     1987
     }                               }
```

Remarque :afin d'alléger l'expression, on emploie généralement le terme enregistrement pour désigner une occurrence d'enregistrement.

Comme nous venons de le voir, chaque champ d'un enregistrement doit avoir un type. Les types de champs autorisés par le logiciel sont soit des types de données prévus par le langage C, soit des des types de données spécifiques au logiciel.

Les types de données prévus par le langage C qui sont reconnus par le logiciel sont :

CHAR, INT, SHORT,
LONG, FLOAT.

Les types de données spécifiques au logiciel sont :

ALPHA, ALPHANUM, NUM.

Le type ALPHA est utilisé pour représenter des données essentiellement alphabétiques et est exprimé grâce au code ISO 5 (la liste des caractères disponibles ainsi que leur représentation est donnée en annexe 2).

Le type ALPHANUM est utilisé pour représenter des données essentiellement alphabétiques ou numériques. Il est exprimé grâce au code ISO 6 (la liste des caractères disponibles ainsi que leur représentation est donnée en annexe 2).

Le type NUM est utilisé pour représenter des données essentiellement numériques et est exprimé grâce au code BCD (la liste des caractères disponibles ainsi que leur représentation est donnée en annexe 2).

Les types ALPHA, ALPHANUM et NUM peuvent être considérés comme des sous-types de CHAR. En effet, toute information exprimable avec les types ALPHA, ALPHANUM et NUM est aussi exprimable avec le type CHAR, mais la réciproque n'est pas vraie.

Ces trois sous-types sont utilisés afin de minimiser l'encombrement des données sur la carte CP8. En effet, s'il faut sept bits (voire même huit) pour représenter un caractère dans le type CHAR (code ASCII), il n'en faut que cinq pour le type ALPHA, six pour le type ALPHANUM et quatre pour le type NUM. Dans la mesure du possible, le concepteur d'application doit donc préférer les types ALPHA, ALPHANUM et NUM au type CHAR dans la définition des types d'enregistrements.

Signalons que le langage C ne reconnaît pas ces trois types de données. Dès lors, afin de permettre l'affectation de valeurs à des champs de l'un de ces types (cfr paragraphe 7.3.9.2.), les types ALPHA, ALPHANUM et NUM ont été rendus compatibles avec le type CHAR. Cela signifie que, dans toute primitive du logiciel, les données de type ALPHA, ALPHANUM et NUM sont déclarées comme des données de type CHAR et donc exprimées en ASCII. Ce n'est que lors d'une écriture ou d'une lecture sur la carte que ces données sont converties.

7.3.9.2. Spécification des opérations réalisables sur un enregistrement

Les opérations réalisables sur un enregistement sont :

- la définition d'un type d'enregistrement,
- la définition d'un champ dans un type d'enregistrement,
- la création d'une occurrence d'enregistrement,
- l'affectation d'une valeur à un champ,
- la lecture de la valeur d'un champ,
- la lecture et l'écriture d'une occurrence d'enregistrement sur une carte,
- la lecture, l'écriture et la suppression d'un type d'enregistrement sur un fichier,
- la lecture et l'écriture d'une occurrence d'enregistrement sur un fichier,
- la saisie et l'affichage d'une occurrence d'enregistrement à l'écran.

a) Définition d'un type d'enregistrement

Fonction : créer un type d'enregistrement ne possédant aucun champ.

Arg : -

Pré : -

Rés : TE : un type d'enregistrement vide.

Post : -

b) Définition d'un champ

Fonction : ajouter à un type d'enregistrement TE un champ NOM-CH de type TYPE-CH.

Arg : TE,
NOM-CH,
TYPE-CH.

Pré : TE est un type d'enregistrement existant.

Rés : RC.

Post : RC a pour valeur :
0 : Le champ NOM-CH a été ajouté au type d'enregistrement TE.
1 : Il existe déjà un champ de nom NOM-CH dans TE.
2 : TYPE-CH est un type de champ invalide.

c) Création d'une occurrence d'enregistrement

Fonction : réserve en mémoire centrale la place nécessaire pour l'occurrence d'un enregistrement de type TE. Cette opération correspond à une déclaration de variable dans les langages évolués.

Arg : TE.

Pré : TE est un type d'enregistrement existant.

Rés : ENREG.

Post : ENREG est un enregistrement de type TE.

d) Affectation d'une valeur à un champ

Fonction : affecter la valeur VAL au champ NOM-CH d'un enregistrement ENREG de type TE.

Arg : TE,
ENREG,
NOM-CH,
VAL.

Pré : TE est un type d'enregistrement existant.
ENREG est un enregistrement existant et son type est TE.
Le type de VAL est compatible avec celui du champ NOM-CH.

Rés : RC.

Post : RC a pour valeur :
0 : La valeur VAL est affectée au champ NOM-CH.
1 : Il n'y a pas de champ NOM-CH dans l'enregistrement ENREG.
2 : VAL contient des caractères non autorisés pour le champ NOM-CH.

e) Lecture de la valeur d'un champ

Fonction : lire la valeur du champ NOM-CH d'un enregistrement ENREG de type TE.

Arg : TE,
ENREG,
NOM-CH.

Pré : TE est un type d'enregistrement existant.
ENREG est un enregistrement existant et son type est TE.

Rés : VAL,
RC.

Post : RC a pour valeur :
0 : VAL est la valeur du champ NOM-CH.
1 : Il n'y a pas de champ NOM-CH dans l'enregistrement ENREG.

f) Ecriture d'un enregistrement sur une carte

Fonction : écrire sur la carte un enregistrement ENREG de type TE à l'adresse spécifiée par ADR et ZONE.

Arg : TE,
ENREG,
ADR,
ZONE.

Pré : TE est un type d'enregistrement existant.
ENREG est un enregistrement existant et son type est TE.
ZONE fait référence à une zone existante.

Rés : RC

Post : RC a pour valeur :
0 : L'enregistrement ENREG est écrit.
Si ADR a une valeur nulle, ENREG est écrit dans le premier emplacement disponible de la zone spécifiée par ZONE (accès séquentiel).
Si ADR a une valeur non nulle, ENREG est écrit dans la zone spécifiée par ZONE à l'adresse ADR (accès direct).
Si ZONE a une valeur nulle, l'adresse spécifiée par ADR est une adresse absolue.
Si ZONE a une valeur non nulle, l'adresse spécifiée par ADR est une adresse relative par rapport au début de la zone spécifiée dans ZONE.
Si ZONE et ADR ont tous deux une valeur nulle, ENREG est écrit dans le premier emplacement libre de la zone de transactions.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : Il n'y a pas assez de place libre pour écrire l'enregistrement.
3 : ADR fait référence à un mot situé en dehors de la zone spécifiée par ZONE (en adressage relatif).
4 : Le mot dont l'adresse est ADR n'est pas libre (en accès direct).
5 : Les conditions d'écriture ne sont pas remplies.
6 : ZONE fait référence à une zone inconnue.
7 : La carte est bloquée.
8 : La carte est invalide.

Remarque - Bien que les zones de service soient situées en zone de transactions, il est interdit d'écrire dans une zone de service si ZONE fait référence à la zone de transactions.

- Les zones connues à ce niveau sont : la zone confidentielle, la zone de lecture, la zone de transactions et les zones de service.

g) Lecture d'un enregistrement d'une carte

Fonction : lire sur la carte l'enregistrement de type TE situé à l'adresse spécifiée par ADR et ZONE.

Arg : TE,
ADR,
ZONE.

Pré : TE est un type d'enregistrement existant.
ZONE fait référence à une zone existante.

Rés : ENREG,
RC.

Post : RC a pour valeur :
0 : ENREG est de type TE.
Si ADR a une valeur nulle, la lecture est séquentielle et ENREG est l'enregistrement qui était courant avant la lecture (après la lecture, l'enregistrement courant est l'enregistrement qui suit ENREG).
Si ADR a une valeur non nulle, ENREG a été lu à l'adresse ADR dans la zone spécifiée par ZONE.
Si ZONE a une valeur nulle, l'adresse spécifiée par ADR est une adresse absolue.
Si ZONE a une valeur non nulle, l'adresse spécifiée par ADR est une adresse relative par rapport au début de la zone spécifiée dans ZONE.
Si ZONE et ADR ont tous deux une valeur nulle, le module considère que ZONE fait référence à la zone de transactions.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : L'enregistrement situé à l'adresse spécifiée par ADR et ZONE n'est pas de type TE.
3 : Le premier mot de l'enregistrement situé à l'adresse spécifiée par ADR et ZONE est un mot libre.
4 : ADR fait référence à un mot situé en dehors de la zone spécifiée par ZONE (en adressage relatif).
5 : Les conditions de lecture ne sont pas remplies.
6 : ZONE fait référence à une zone inconnue.
7 : La carte est bloquée.

Remarque : Les zones connues à ce niveau sont : la zone confidentielle, la zone de lecture, la zone de transactions et les zones de service.

h) Ecriture d'un type d'enregistrement dans un fichier

Fonction : écrire le type d'enregistrement TE sous le nom NOM-TE à la fin du fichier FICH.

Arg : TE,
NOM-TE,
FICH.

Pré : TE est un type d'enregistrement existant.

Rés : RC.

Post : RC a pour valeur :

0 : TE est écrit sous le nom NOM-TE à la fin du fichier FICH. Si le fichier FICH n'existe pas, il est créé par le module.
1 : Il y a déjà un type d'enregistrement de nom NOM-TE dans le fichier FICH.

i) Lecture d'un type d'enregistrement d'un fichier

Fonction : lire le type d'enregistrement de nom NOM-TE dans le fichier FICH.

Arg : NOM-TE,
FICH.

Pré : -

Rés : TE,
RC.

Post : RC a pour valeur :
0 : TE est le type d'enregistrement de nom NOM-TE situé dans le fichier FICH.
1 : Il n'y a pas de type d'enregistrement de nom NOM-TE dans le fichier FICH.
2 : Le fichier FICH n'existe pas.

j) Suppression d'un type d'enregistrement d'un fichier

Fonction : supprimer le type d'enregistrement de nom NOM-TE du fichier FICH.

Arg : NOM-TE,
FICH.

Pré : -

Rés : RC.

Post : RC a pour valeur :
0 : Le type d'enregistrement de nom NOM-TE a été supprimé du fichier FICH.
1 : Il n'y a pas de type d'enregistrement de nom NOM-TE dans le fichier FICH.
2 : Le fichier FICH n'existe pas.

k) Ecriture d'un enregistrement dans un fichier

Fonction : écrire l'enregistrement ENREG de type TE à la fin du fichier FICH.

Arg : TE,
ENREG,
FICH.

Pré : TE est un type d'enregistrement existant.
ENREG est un enregistrement existant et son type est TE.

Rés : -

Post : -

l) Lecture d'un enregistrement d'un fichier

Fonction : lire l'enregistrement courant de type TE dans le fichier FICH.

Arg : TE,
FICH.

Pré : TE est un type d'enregistrement existant.
FICH est ouvert en lecture.

Rés : ENREG,
RC.

Post : RC a pour valeur :
0 : ENREG est l'enregistrement courant du fichier FICH au début de la lecture. Son type est TE. A la fin de la lecture, l'enregistrement courant devient l'enregistrement qui suit ENREG.
1 : L'enregistrement courant n'est pas de type TE.
2 : Fin du fichier.

m) Saisie d'un enregistrement à l'écran

Fonction : saisit des valeurs à l'écran et les affecte aux champs d'un enregistrement de type TE.

Arg : TE.

Pré : TE est un type d'enregistrement existant.

Rés : ENREG,
RC.

Post : RC a pour valeur :
0 : ENREG est un enregistrement de type TE dont les champs contiennent les valeurs saisies à l'écran.
1 : Il existe des champs de ENREG qui n'ont pas reçu de valeur.

n) Affichage d'un enregistrement à l'écran

Fonction : afficher à l'écran le nom et le contenu des champs d'un enregistrement ENREG de type TE.

Arg : TE,
ENREG.

Pré : TE est un type d'enregistrement existant.
ENREG est un enregistrement existant et son type est TE.

Rés : -

Post : -

Recommandations de réalisation

Comme l'enregistrement est une structure de données essentiellement dynamique, une structure de liste convient bien pour le représenter.

Un type d'enregistrement peut se représenter de la manière suivante :

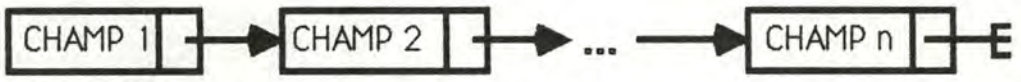


Fig 7.4 : Représentation d'un type d'enregistrement

Où un champ est constitué d'un nom de champ et d'un type.

Une occurrence peut se représenter de la manière suivante :

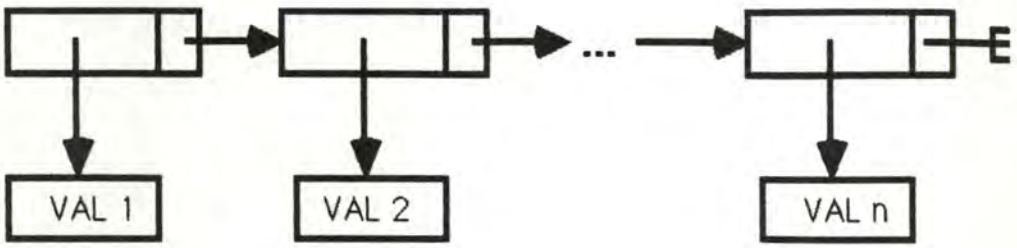


Fig 7.5 : Représentation d'une occurrence d'enregistrement

Où VAL i est une valeur dont le type est spécifié dans CHAMP i.

7.3.10. Module 2.2 : Mot

Rappelons qu'un mot est un ensemble de 32 bits qui inclut des bits systèmes.

7.3.10.1. Ecriture d'un mot sur une carte

Fonction : écrire le mot MOT sur la carte à l'adresse spécifiée par ADR et ZONE.

Arg : MOT,
ADR,
ZONE.

Pré : ZONE fait référence à une zone existante.

Rés : RC.

Post : RC a pour valeur :

0 : Le mot est écrit à l'adresse spécifiée par ADR et ZONE. Si le premier bit de mot (bit V) vaut zéro, alors l'écriture est validée.
Si ADR a une valeur nulle, MOT est écrit dans le premier emplacement disponible de la zone spécifiée par ZONE (accès séquentiel).
Si ADR a une valeur non nulle, MOT est écrit dans la zone spécifiée par ZONE à l'adresse ADR (accès direct).
Si ZONE a une valeur nulle, l'adresse spécifiée par ADR est une adresse absolue.
Si ZONE a une valeur non nulle, l'adresse spécifiée par ADR est une adresse relative par rapport au début de la zone spécifiée dans ZONE.
Si ZONE et ADR ont tous deux une valeur nulle, MOT est écrit dans le premier emplacement libre de la zone de transactions.

1 : Il n'y a pas de carte dans le lecteur-encodeur.

2 : Il n'y a pas assez de place libre pour écrire le mot.

3 : ADR fait référence à un mot situé en dehors de la zone spécifiée par ZONE (en adressage relatif).

4 : Le mot dont l'adresse est ADR n'est pas libre (en accès direct).
5 : Les conditions d'écriture ne sont pas remplies.
6 : ZONE fait référence à une zone inconnue.
7 : La carte est bloquée.
8 : La carte est invalide.

Remarque : Les zones connues à ce niveau sont : la zone des clés porteurs, la zone confidentielle, la zone de lecture, la zone de transactions, la zone de fabrication et les zones de service.

7.3.10.2. Lecture d'un mot sur une carte

Fonction : lire sur la carte le mot dont l'adresse est spécifiée par ADR et ZONE.

Arg : ADR,
ZONE.

Pré : ZONE fait référence à une zone existante.

Rés : MOT,
RC.

Post : RC a pour valeur :
0 : MOT est le contenu du mot situé à l'adresse spécifiée par ADR et ZONE.
Si ADR a une valeur nulle, la lecture est séquentielle et MOT est le mot qui était courant avant la lecture (après la lecture, le mot courant est le mot qui suit MOT).
Si ADR a une valeur non nulle, MOT a été lu à l'adresse ADR dans la zone spécifiée par ZONE.
Si ZONE a une valeur nulle, l'adresse spécifiée par ADR est une adresse absolue.
Si ZONE a une valeur non nulle, l'adresse spécifiée par ADR est une adresse relative par rapport au début de la zone spécifiée dans ZONE.
Si ZONE et ADR ont tous deux une valeur nulle, le module considère que ZONE fait référence à la zone de transactions.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : ADR fait référence à un mot situé en dehors de la zone spécifiée par ZONE (en adressage relatif).
3 : Les conditions de lecture ne sont pas remplies.
4 : ZONE fait référence à une zone inconnue.
5 : La carte est bloquée.

Remarque : Les zones connues à ce niveau sont : la zone d'accès, la zone confidentielle, la zone de lecture, la zone de transactions, la zone de fabrication et les zones de service.

7.3.11. Module 2.3 : Diversification

Fonction : diversifier la clé de fabrication, les clés émetteurs, la clé interne et la clé porteur de type 1.

Arg : REF-CLE : référence de la clé à diversifier.

Pré : -

Rés : CLE,
RC.

Post : RC a pour valeur :
0 : CLE est la clé diversifiée correspondant à REF-CLE.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : REF-CLE ne correspond à aucune clé.

7.3.12. Module 2.4 : Contrôle de parité

Fonction : calcule les bits de parité correspondant aux 27 bits de poids fort du mot MOT.

Arg : MOT.

Pré : -

Rés : PARITE.

Post : PARITE est un octet dont les cinq bits de poids faible sont les bits de parité correspondant aux 27 bits de poids fort du mot MOT.

7.3.13. Module 3.1 : Interface écran

Ce module réalise deux fonctions complémentaires aux ordres de lecture standards du langage C.

7.3.13.1. Saisie d'une valeur de type donné

Fonction : saisir une valeur d'un type TYPE spécifique au logiciel.

Arg : TYPE.

Pré : -

Rés : VAL,
RC.

Post : RC a pour valeur :
0 : VAL contient la valeur saisie et son type est TYPE.
1 : TYPE est un type de données non spécifique au logiciel.

7.3.13.2. Saisie d'une valeur comprise entre deux limites

Fonction : saisir une valeur numérique comprise entre deux limites LIM-INF et LIM-SUP.

Arg : LIM-INF,
LIM-SUP.

Pré : 0 ≤ LIM-INF ≤ LIM-SUP.

Rés : VAL,
RC.

Post : RC a pour valeur :
0 : LIM-INF ≤ VAL ≤ LIM-SUP.
1 : VAL > LIM-SUP.
2 : VAL < LIM-INF.

7.3.14. Module 3.2 : Conversion de codes

7.3.14.1. ASCII -> ISO 5

Fonction : transformer une donnée SOURCE exprimée en ASCII en une donnée CIBLE exprimée en ISO 5.

Arg : SOURCE.

Pré : SOURCE est exprimée en ASCII.

Rés : CIBLE,
RC.

Post : RC a pour valeur :

0 : CIBLE est l'équivalent de SOURCE exprimé en ISO 5.
1 : SOURCE contient des caractères intraduisibles en ISO 5.

7.3.14.2. ASCII -> ISO 6

Fonction : transformer une donnée SOURCE exprimée en ASCII en une donnée CIBLE exprimée en ISO 6.

Arg : SOURCE.

Pré : SOURCE est exprimée en ASCII.

Rés : CIBLE,
RC.

Post : RC a pour valeur :
0 : CIBLE est l'équivalent de SOURCE exprimé en ISO 6.
1 : SOURCE contient des caractères intraduisibles en ISO 6.

7.3.14.3. ASCII -> BCD

Fonction : transformer une donnée SOURCE exprimée en ASCII en une donnée CIBLE exprimée en BCD.

Arg : SOURCE.

Pré : SOURCE est exprimée en ASCII.

Rés : CIBLE,
RC.

Post : RC a pour valeur :
0 : CIBLE est l'équivalent de SOURCE exprimé en BCD.
1 : SOURCE contient des caractères intraduisibles en BCD.

7.3.14.4. ISO 5 -> ASCII

Fonction : transformer une donnée SOURCE exprimée en ISO 5 en une donnée CIBLE exprimée en ASCII.

Arg : SOURCE.

Pré : SOURCE est exprimée en ISO 5.

Rés : CIBLE.

Post : CIBLE est l'équivalent de SOURCE exprimé en ASCII.

7.3.14.5. ISO 6 -> ASCII

Fonction : transformer une donnée SOURCE exprimée en ISO 6 en une donnée CIBLE exprimée en ASCII.

Arg : SOURCE.

Pré : SOURCE est exprimée en ISO 6.

Rés : CIBLE.

Post : CIBLE est l'équivalent de SOURCE exprimé en ASCII.

7.3.14.6. BCD -> ASCII

Fonction : transformer une donnée SOURCE exprimée en BCD en une donnée CIBLE exprimée en ASCII.

Arg : SOURCE.

Pré : SOURCE est exprimée en BCD.

Rés : CIBLE.

Post : CIBLE est l'équivalent de SOURCE exprimé en ASCII.

7.3.15. Module 3.3 : Gestion des adresses

7.3.15.1. Adresse absolue -> adresse relative

Fonction : convertir l'adresse absolue ADR-A en une adresse relative à la zone ZONE.

Arg : ADR-A,
ZONE.

Pré : ZONE fait référence à une zone existante.

Rés : ADR-R,
RC.

Post : RC a pour valeur :
0 : ADR-R est l'adresse relative à la zone ZONE correspondant à l'adresse absolue ADR-A.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : ADR-A n'est pas une adresse valide.
3 : ZONE fait référence à une zone inconnue.

Remarque : Les zones connues à ce niveau sont : la zone des clés porteurs, la zone d'accès, la zone confidentielle, la zone de lecture, la zone de transactions, la zone de fabrication et les zones de service.

7.3.15.2. Adresse relative -> adresse absolue

Fonction : convertir l'adresse ADR-R relative à la zone ZONE en une adresse absolue.

Arg : ADR-R,
ZONE.

Pré : ZONE fait référence à une zone existante.

Rés : ADR-A,
RC.

Post : RC a pour valeur :
0 : ADR-A est l'adresse absolue correspondant à l'adresse ADR-R relative à la zone ZONE.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : ADR-R n'est pas une adresse valide.
3 : ZONE fait référence à une zone inconnue.

Remarque : Les zones connues à ce niveau sont : la zone des clés porteurs, la zone d'accès, la zone confidentielle, la zone de lecture, la zone de transactions, la zone de fabrication et les zones de service.

7.3.15.3. Recherche de mot libre

Fonction : rechercher l'adresse du premier mot libre de la zone ZONE.

Arg : ZONE.

Pré : ZONE fait référence à une zone existante.

Rés : ADR,
RC.

Post : RC a pour valeur :
0 : ADR est l'adresse du premier mot libre de la zone ZONE. Si ZONE a une valeur nulle, ADR est l'adresse du premier mot libre de la carte.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : Il n'y a pas de mot libre dans la zone ZONE.
3 : Les conditions de lecture ne sont pas remplies.
4 : La carte est bloquée.
5 : ZONE fait référence à une zone inconnue.

Remarque : Les zones connues à ce niveau sont : la zone d'accès, la zone confidentielle, la zone de lecture, la zone de transactions, la zone de fabrication et les zones de service.

7.3.15.4. Recherche d'une zone de service

Fonction : rechercher l'adresse du premier mot d'une zone de service de référence REF.

Arg : REF.

Pré : La carte est personnalisée.

Rés : ADR,
RC.

Post : RC a pour valeur :
0 : ADR est l'adresse du premier mot de la zone de service dont la référence est REF. Cette adresse est une adresse relative par rapport au début de la zone de transactions
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : La zone de service de référence REF n'existe pas sur la carte.
3 : Les conditions de lecture ne sont pas remplies.
4 : La carte est bloquée.

7.3.16. Module 4.1 : Interface carte

7.3.16.1. Mise sous tension

Fonction : mettre la carte sous tension.

Arg : -

Pré : -

Rés : PE,
PL,
EC,
RC.

Post : RC a pour valeur :
0 : La carte est sous tension.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : Le masque de la carte insérée n'est pas un masque M4.
3 : Un incident matériel s'est produit.
4 : La zone d'accès a enregistré la présentation d'une clé porteur fausse.
5 : La zone d'accès a enregistré la présentation de deux clés porteurs fausses.
6 : La carte est bloquée.
7 : La zone d'accès est saturée.
8 : La carte est invalide.

PE a pour valeur :
0 : La zone de transactions est en écriture libre.
1 : La zone de transactions est en écriture protégée.

PL a pour valeur :
0 : La zone de transactions est en lecture libre.
1 : La zone de transactions est en lecture protégée.

EC a pour valeur :
0 : La carte n'est pas personnalisée.
1 : La carte est personnalisée.

PE, PL et EC ne sont significatifs que si RC vaut 0, 4, 5 ou 6.

7.3.16.2. Mise hors tension

Fonction : mettre la carte hors tension.

Arg : -

Pré : -

Rés : -

Post : -

7.3.16.3. Ecriture de mot

Fonction : écrire le mot MOT à l'adresse ADR de la zone ZONE.

Arg : MOT,
ADR,
ZONE.

Pré : ZONE fait référence à une zone existante.

Rés : RC.

Post : RC a pour valeur :
0 : Le mot MOT est écrit à l'adresse ADR dans la zone ZONE.
Si le bit de validation de MOT est positionné, le mot écrit est aussi validé et le contenu de ce mot est égal à MOT.
Si le bit de validation n'est pas positionné, MOT est écrit mais non validé. Dans ce cas, l'information écrite peut être différente du contenu de MOT.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : Les conditions d'écriture ne sont pas remplies.
3 : ZONE fait référence à une zone inconnue.
4 : Un incident est survenu en écriture.
5 : La carte est bloquée.
6 : La carte est invalide.

Remarque : Les zones connues à ce niveau sont : la zone des clés porteurs, la zone confidentielle, la zone de lecture, la zone de transactions et la zone de fabrication.

7.3.16.4. Validation de mot

Fonction : valider le mot situé à l'adresse ADR de la zone ZONE.

Arg : ADR,
ZONE.

Pré : ZONE fait référence à une zone existante.

Rés : RC.

Post : RC a pour valeur :
0 : Le mot écrit à l'adresse ADR dans la zone ZONE est validé.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : Les conditions de validation ne sont pas remplies.
3 : ZONE fait référence à une zone inconnue.
4 : Un incident est survenu en validation.
5 : La carte est bloquée.
6 : La carte est invalide.

Remarque : Les zones connues à ce niveau sont : la zone des clés porteurs, la zone confidentielle, la zone de lecture, la zone de transactions et la zone de fabrication.

7.3.16.5. Lecture de mot

Fonction : lire le mot situé à l'adresse ADR dans la zone ZONE.

Arg : ADR,
ZONE.

Pré : ZONE fait référence à une zone existante.

Rés : MOT,
RC.

Post : RC a pour valeur :
0 : MOT a pour valeur le contenu du mot situé à l'adresse ADR dans la zone ZONE.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : Les conditions de lecture ne sont pas remplies.
3 : ZONE fait référence à une zone inconnue.
4 : La carte est bloquée.

Remarque : Les zones connues à ce niveau sont : la zone d'accès, la zone confidentielle, la zone de lecture, la zone de transactions et la zone de fabrication.

7.3.16.6. Génération de certificat

Fonction : générer un certificat en prenant comme paramètres la clé interne de la carte, l'adresse ADR, le contenu du mot dont l'adresse est ADR et le message d'entrée E.

Arg : E,

ADR.

Pré : -

Rés : R,
RC.

Post : RC a pour valeur :
0 : R est le certificat obtenu en activant Télépass avec comme paramètres la clé interne de la carte, l'adresse ADR, le contenu du mot dont l'adresse est ADR et le message d'entrée E.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : Les conditions de lecture ne sont pas remplies.
3 : La carte est bloquée.

7.3.16.7. Présentation de clé

Fonction : présenter une clé CLE de type TYPE à la carte et la valider en lecture.

Arg : CLE,
TYPE.

Pré : TYPE désigne la clé de fabrication, une clé porteur, la clé émetteur primaire ou la clé émetteur secondaire.

Rés : RC.

Post : RC a pour valeur :
0 : La clé est correcte.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : La zone d'accès a enregistré la présentation d'une clé porteur ou d'une clé de fabrication fausse.
3 : La zone d'accès a enregistré la présentation de deux clés porteurs ou de deux clés de fabrication fausses.
4 : La carte est bloquée.
5 : La zone d'accès est saturée.

7.3.16.8. Déblocage

Fonction : présenter la clé porteur CLE-P et la clé émetteur primaire CLE-EP à la carte et les valider en lecture.

Arg : CLE-P,
CLE-EP.

Pré : -

Rés : RC.

Post : RC a pour valeur :
0 : Les clés sont correctes.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : La clé porteur présentée est fausse.
3 : La clé émetteur primaire présentée est fausse.
4 : Les deux clés présentées sont fausses.
5 : La zone d'accès est saturée.

7.3.16.9. Ecriture de verrou

Fonction : positionner le verrou V.

Arg : V.

Pré : -

Rés : RC.

Post : RC a pour valeur :
0 : Le verrou V est positionné.
1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : V désigne un verrou inexistant.

7.3.16.10. Adresse relative -> adresse absolue

Fonction : convertir l'adresse ADR-R relative à la zone ZONE en une adresse absolue.

Arg : ADR-R,
ZONE.

Pré : ZONE fait référence à une zone existante.

Rés : ADR-A,
RC.

Post : RC a pour valeur :
0 : ADR-A est l'adresse absolue correspondant à l'adresse ADR-R relative à la zone ZONE.

1 : Il n'y a pas de carte dans le lecteur-encodeur.
2 : ADR-R n'est pas une adresse valide.
3 : ZONE fait référence à une zone inconnue.

Remarque : Les zones connues à ce niveau sont : la zone des clés porteurs, la zone d'accès, la zone confidentielle, la zone de lecture, la zone de transactions, la zone de fabrication et la zone des verrous.

CONCLUSION

Au terme de notre étude, il ressort que la carte à microprocesseur Masque 4 développée par BULL CP8 est un produit globalement satisfaisant. Deux points forts plaident en sa faveur. Ce sont, d'une part, un niveau de protection de l'information extrêmement élevé et, d'autre part, un très haut degré de polyvalence (cfr chapitre 4).

A côté de ces deux qualités fondamentales, trois points faibles méritent d'être rappelés. Tout d'abord, la sécurité des clés utilisées pour la diversification n'est pas absolue (cfr paragraphe 4.1.2.2. La diversification). Ensuite, l'indépendance entre les différents services accessibles au moyen de la carte n'est que partielle (cfr paragraphe 4.3.2.). Enfin, la programmation d'applications basées sur la carte est loin d'être aisée à cause de l'absence d'interface entre la carte et les langages de programmation évolués (cfr paragraphe 5.1.1.). Cette faiblesse doit cependant être relativisée puisque des logiciels d'aide à la programmation d'applications, tels que celui présenté dans la seconde partie de ce mémoire, commencent à voir le jour sous l'impulsion des sociétés de services.

Pour clôturer cette évaluation, il faut encore signaler que la carte Masque 4 est l'une des premières à avoir été développées par BULL CP8 et que les cartes actuellement en cours d'étude ont été considérablement améliorées : la sécurité a encore été accrue, les services sont devenus totalement indépendants et la programmation d'applications est simplifiée grâce à l'apparition de nouveaux langages dédicacés à la gestion de la carte (voir à ce sujet le mémoire d'Yves Triniane).

BIBLIOGRAPHIE

Ce mémoire a été entièrement réalisé sur base de documentation interne à la société BULL CP8. Tout renseignement complémentaire sur la carte à microprocesseur qui y est présentée pourra être obtenu en contactant

BULL CP8
rue Eugène Henaff

78190 Trappes
France

ANNEXE 1

NORMES ISO POUR LE DIALOGUE AVEC LES CARTES A MICROPROCESSEUR

ISO/TC 97/SC 17/WG 4 N 260

DRAFT PROPOSAL ISO/DP 7816/3	
date 1986-10-09	ISO/TC 97/SC 17
superseded document 97/17/4 N 244	

THIS DOCUMENT IS STILL UNDER STUDY AND SUBJECT TO CHANGE. SHOULD NOT BE USED FOR REFERENCE PURPOSES.

ISO/TC 97/SC 17 "IDENTIFICATION AND CREDIT CARDS" Secretariat B.S.I.	Circulated to P- and O-members, technical committees and international organizations in liaison for : – discussion at – comments by – voting by (P-members only)

Title IDENTIFICATION CARDS - INTEGRATED CIRCUIT(S) CARD WITH CONTRACTS

PART 3 : ELECTRONIC SIGNALS AND EXCHANGE PROTOCOLS.

Introductory note

049 E-79.05-500

IDENTIFICATION CARDS - INTEGRATED CIRCUIT(S) CARD WITH CONTACTS-
PART 3 : ELECTRONIC SIGNALS AND EXCHANGE PROTOCOLS.

0 - INTRODUCTION

This International Standard is one of a series of standards describing the parameters for integrated circuit cards with contacts and the use of such cards for international interchange. These IC cards are identification cards intended for transactions negotiated between the outside and the integrated circuit in the card. As a result of a transaction, the card delivers information (computation results, stored data), and/or modifies its content (data storage, event memorisation).

1 - SCOPE AND FIELD OF APPLICATION

This part 3 of ISO 7816 specifies the power and signal structures, and data and command structures for a preliminary dialogue between an integrated circuit(s) card and an interface device such as a terminal.

It will cover power, signal rates and voltage levels, parity conventions, operation procedures, transmission mechanisms, provisions for naming applications, additional aspects of data interchange format standardization and the organisation of communication with the IC card. It does not cover information and instruction content, such as identification of issuers and users, services and limits, security features, journaling, and those instruction definitions and instruction processing, which are defined in existing standards and in additional standards to be developed.

Illustration of the specifications of this standard is shown in annex 1 which does not form part of this standard.

2 - REFERENCES

ISO/ 1177- " Information processing- Character structure for start/stop and synchronous transmission".

ISO/7810 - "Identification cards - Physical characteristics"

ISO/DIS 7816/1 - "Identification Cards - Integrated circuit(s) card with contacts - Part 1 : Physical characteristics"

ISO/DIS 7816/2.2 - "Identification Cards - Integrated circuit(s) card with contacts - Part 2 : Dimensions and location of the contacts".

3 - DEFINITIONS

The term identification card is defined in ISO/DIS 7810.

Interface device : a terminal, communication device or machine to which the integrated circuit card is electrically connected during operation.

4 - ELECTRICAL CHARACTERISTICS OF THE CONTACTS

4.1 Electrical Functions

Contacts assignments are specified in part 2 of ISO 7816, supporting at least the following electrical circuits:

I/O : Circuit by which serial data is input to or output from the integrated circuit inside the card.

VPP : The programming supply voltage Vpp (optional use by the card).

GND : Zero voltage as a reference voltage (0 V).

CLK : Clocking or timing signal (optional use by the card).

RST : Circuit either used by itself (reset signal) or in combination with an additional control circuit to define specific function modes (optional use by the card, provided external Vcc supplies the reset).

VCC : The circuit supply voltage (optional use by the card).

Note : The use of two remaining contacts should be defined in the appropriate application standards and submitted to ISO TC 97/SC 17 for approval.

4.2 Voltage and Current Values

All measurements are defined with respect to contact GND and in a temperature range of 0°C to 50°C.

All currents flowing into the card are considered positive.

All timings shall be measured relative to the appropriate treshold levels as defined in 4.2.2 to 4.2.5.

4.2.1 Abbreviations used

V_{IH} stands for high level input voltage

V_{IL} stands for low level input voltage

I_{CC} stands for supply current

I_{IH} stands for high level input current

I_{IL} stands for low level input current

V_{OH} stands for high level output voltage

V_{OL} stands for low level output voltage

I_{OH} stands for high level output current

I_{OL} stands for low level output current

C_{IN} input capacitance

C_{OUT} output capacitance

t_R rise time, between 10% and 90% of signal amplitude.

t_F fall time, between 90% and 10% of signal amplitude.

4.2.2 I/O

This contact is used for data exchange as input or output.

Electrical Characteristics

SYMBOL	CONDITIONS		MIN	MAX	UNIT
V_{IH}	Either	I_{IH} max = - 500 uA	2	Vcc	V
	Or(1)	I_{IH} max = ± 20 uA	0.7 * Vcc	Vcc - 0,3	V
V_{IL}		I_{IL} max = - 1mA	- 0,3	0,8	V
V_{OH} (2)	Either	I_{OH} max = - 100 uA	2,4	Vcc	V
	or	I_{OH} max = ± 20 uA	3,8	Vcc	V
V_{OL}		I_{OL} max = 1 mA	0	0,4	V
C_{IN}, C_{OUT}				30	pF
t_R , t_F				1	us

(1) For the interface device, take into account both conditions.

(2) It is assumed that a pull up resistor is used in the interface device (recommended value: 20 K ohms)

When the two ends of the line are in reception mode, the line shall be maintained in the high state.

When the two ends are in non-matched transmit mode, the logic state of the line may be indeterminate. During operation, the interface device and the card shall not both be in transmit mode.

Note: I/O thus has two steady states (as defined in ISO 1177):

- state Z (mark) or high state, if the card and the interface device are in reception mode or if this state is imposed by the transmitter.

- state A (space) or low state, if this state is imposed by the transmitter.

4.2.3 VPP

Idle state : Vcc ± 5 %, 20 mA max.

Active state : P volts ± pa %, I value: See note 3.
pa = programming voltage accuracy, see 6.1.4.3.

Rise or fall time : 200 us max. The rate of change of VPP shall not exceed 2 volts/us.

Note 1: the card provides the interface device with the values of P, pa and I. Default values are P= 25 V, pa = 4 %, I max = 50 mA.

Note 2: the measurements of these figures shall be made in the range of 25 ± 5°C

Note 3: maximum values of I are the following:

00	25 mA
01	50 mA
10	100 mA
11	200 mA

4.2.4 - CLK

The actual frequency, delivered by the interface device on CLK, is designated by f_i or f_s. It may not be the same during the answer to reset (f_i) and during subsequent transmission (f_s).

Electrical Characteristics

SYMBOL	CONDITIONS	MIN	MAX	UNIT
V_{IH}	I_{IH} max = - 200 uA	2,4	Vcc + 0,3	V
	I_{IH} max = ± 20 uA	0,7 * Vcc	Vcc + 0,3	V
	I_{IH} max = ± 10 uA	Vcc - 0,7	Vcc + 0,3	V
V_{IL}	I_{IL} max = ± 200 uA	- 0,3	0,5	V
C_{IN}			30	pF
t_R, t_F			9% of period with a maximum of 0,5 us.	

Duty cycle for asynchronous operation : between 45% and 55 % of the period during stable operation.

4.2.5 RST

Electrical Characteristics

SYMBOL	CONDITIONS	MIN	MAX	UNIT
V_{IH}	I_{IH} max = - 200 uA	4	Vcc + 0,3	V
V_{IH}	I_{IH} max = ± 10 uA	Vcc - 0,7	Vcc + 0,3	V
V_{IL}	I_{IL} max = ± 200 uA	- 0,3	0,6	V

4.2.6 - VCC

5 V ± 5 %.

Icc, the current consumed by the card, shall remain less than 200 mA.

Note: When Vcc is used for programming instead of Vpp, Icc current shall be less than 300 mA.

5 - OPERATING PROCEDURE FOR IC CARDS

This operating procedure is applicable for every integrated circuit card with contacts.

A transaction with the card is conducted through the consecutive operations:

- Activation of the contacts
- Reset of the card
- Answer to Reset
- Command(s)
- Deactivation of the contacts

Notes: 1. Reset of a card can be initiated by the interface device at its discretion at any time.

2. A connector is inactive when all pins remain between 0 V and 0.4 V, referenced to GND pin for currents less than 1 mA.

3. An active state on VPP shall only be provided and maintained when requested by the card.

5.1 Activation of the Contacts

The electrical circuits shall not be activated until the contacts are positioned with respect to the interface device so as to avoid possible damage to any card meeting these standards.

The activation of the contacts by the interface device consists of the consecutive operations:

RST is in low state, VCC shall be powered, I/O in the interface device shall be put in reception mode, VPP shall be raised to idle state, and CLK shall be provided with a suitable and stable clock (see 4.2.4 - CLK).

5.2 Reset of the Card

See figure 1 and figure 2.

By the end of the activation of the contacts (RST in low state, VCC powered and stable, I/O in reception mode in the interface device, VPP stable at idle state, CLK provided with a suitable and stable clock), the card answering asynchronously is ready for reset:

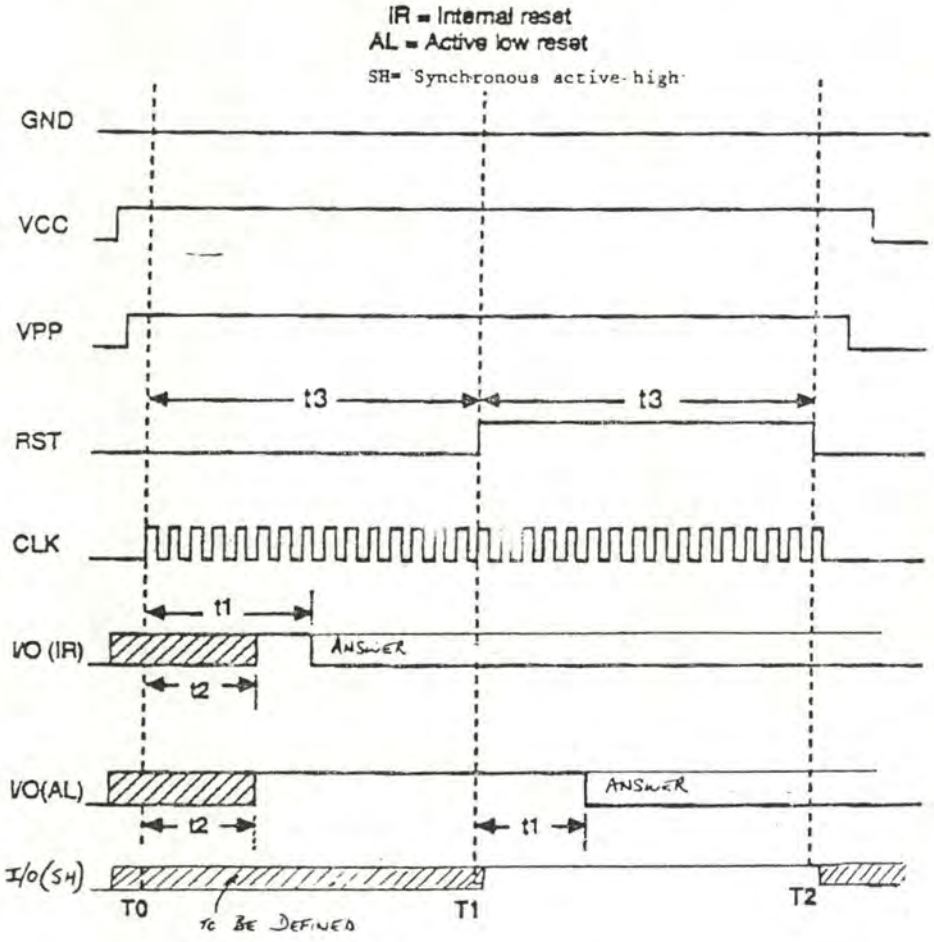
The clock signal is applied to CLK at time T0. The I/O line shall be set to a defined state (high-Z) within 200 clock cycles of the clock signal being applied to CLK (time t2 after T0).

A card with an internal reset is reset after a few cycles of the clock signal. Thus, if the reset is internal, the answer on I/O shall begin between 400 and 40 000 clock cycles after the clock signal is applied to CLK (time t1 after T0).

A card with an active low reset is reset by maintaining RST in low state for at least 40 000 clock cycles after the clock signal is applied to CLK (time t3 after T0). Thus, if no answer occurs within the last 40 000 clock cycles with RST in low state, RST is put to high state (at time T1). The answer on I/O shall begin between 400 and 40 000 clock cycles after the rising edge of the signal on RST (time t1 after T1).

If no answer has been received after 40 000 clock cycles with RST in high state (t3 after T1), the signal on RST shall be returned to low state (at time T2) and the contacts shall be deactivated.

Figure 1 : Examples of reset of the card - Asynchroncus Transmission.



$400 \cdot 1/f \leq t1 \leq 40{,}000 \cdot 1/f$

$t2 \leq 200 \cdot 1/f$

$t3 \geq 40{,}000 \cdot 1/f$

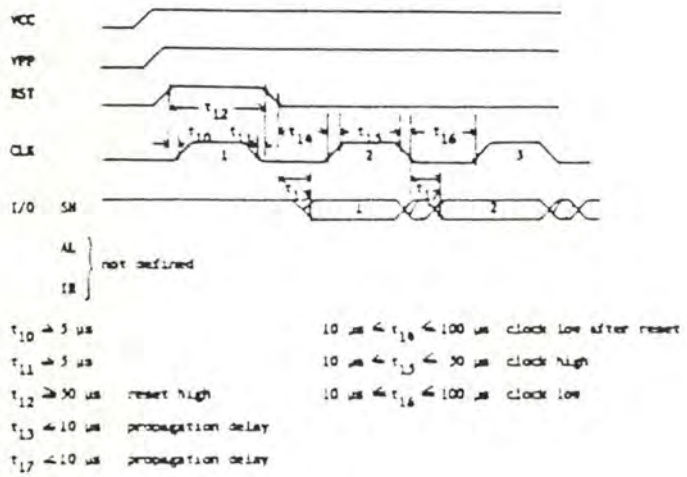
note: the hatched area indicates a period when the state of I/O is undefined .

With a card answering synchronously, the interface device sets all lines to low state. Then VCC is powered, VPP is set to idle state, CLK and RST remain in low state, I/O is put in reception mode in the interface device. A high state shall be maintained on RST for at least 50 microseconds (t_{12})before returning to low state again (see figure 2).

The clock pulse is applied after an interval (t_{10}) from the rising edge of the reset signal. The duration for the high state of the clock pulse can be any value between 10 us and 50 us and there is only no more than one clock pulse during reset high allowed. The time interval between the falling edges of CLK and RST is (t_{11}).

The first data bit is obtained as an answer on I/O while CLK is low and is valid after an interval (t_{13}) from the falling edge on RST.

Figure 2 Example when a synchronous answer is expected



Note 1 : The internal state of the card is not defined before reset. Therefore the design of the IC card has to avoid improper operation.

Note 2 : In order to continue the transaction with the card, RST shall be maintained in the state where an answer occurs on I/O.

Note 3 : Interface devices may support one or more of these types of reset behaviour. The priority of testing for asynchronous or synchronous cards is not defined in this standard.

5.3 Command(s)

All data exchanged over the I/O circuit correspond to the execution of commands (via RST for reset and via I/O for any other command).

During the processing of a command, the wait for a signal shall not last longer than a maximum delay, named waiting time. Otherwise it shall be assumed that the card or the interface device is unresponsive, showing its disapproval for instance.

As for answer to reset, the operating procedure of commands depend on the type of transmission (asynchronous or synchronous).

5.4 Deactivation of the contacts

When the processing of the last command of the sequence is complete, or when the transaction is aborted (unresponsive card or detection of card removal), the electrical contacts shall be deactivated.

Note: The deactivation by the interface device consists of the consecutive operations: low state on RST, low state on CLK, power off on VPP, state A on I/O, power off on VCC.

6 - ANSWER TO RESET

Two types of transmission are considered:

Asynchronous Transmission

In this type of transmission, characters are transmitted on the I/O line in an asynchronous half duplex mode. Each character codes an 8 bit byte.

Synchronous Transmission

In this type of transmission, a series of bits is transmitted on the I/O line in an half-duplex mode in synchronisation with the clock signal on CLK.

6.1 Answer to reset in an Asynchronous Transmission

6.1.1 Bit Duration

There is a linear relationship between the Elementary Time Unit (etu) used on I/O and the period provided by the interface device on CLK.

$$\text{initial etu} = \frac{f_o}{f_i} \cdot \frac{1}{9600} \text{ (seconds)}$$ see alternate definition of etu in 6.1.4.1.

f_o = reference frequency (characteristic of a card) in MHz

f_i = initial frequency (provided by the interface device during the answer to reset) in MHz, as defined in 4.2.4.

f_o is the reference frequency required by the card to generate 9600 bits/s.
f_o = 3,579545 MHz.

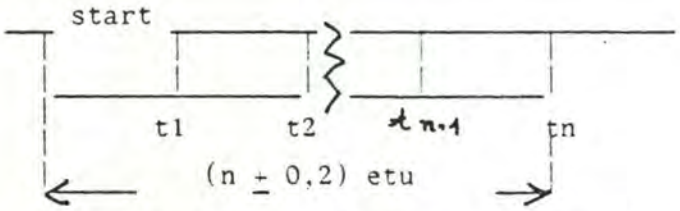
In order to read the initial character (TS), all cards shall initially be operated with f_i in the range of 1 to 4 MHz.

6.1.2 Character Frame

Prior to the transmission of a character, I/O shall be in state Z. A character consists of ten consecutive bits: a start bit in state A, eight bits of information, designated ba to bh and conveying a data byte, and a tenth bit bi used for even parity checking.

Within a character, the time from the leading edge of the start bit to the trailing edge of the nth bit shall equal (n ± 0,2) etu.

The delay between two consecutive characters (between start leading edges) is at least 12 etu, including a character duration (10 ± 0,2) etu plus a guardtime. While in guardtime, the interface device and the card remain both in reception, so that I/O is in state Z.



A data byte consists of 8 bits designated b1 to b8, from the least significant bit (lsb, b1) to the most significant bit (msb, b8). Conventions (level coding, connecting levels Z/A to digits (1 or 0) ; and bit significance, connecting ba - bh to b1 - b8) are specified in the first character, called TS, which is transmitted by the card in response to reset. Parity is correct when the number of ONES is even in the sequence from ba to bi.

During the answer to reset, the delay between two consecutive characters from the card shall not exceed 9600 etu, which is equivalent to f_o/f_i seconds.

This maximum value of the delay during answer to reset is named initial waiting time.

6.1.3 Use of Error Detection and Character Repetition

The transmitter tests I/O, (11 ± 0,2) etu after the start leading edge:

- If I/O is in state Z, the correct reception is assumed.
- If I/O is in state A, the transmission was incorrect. The disputed character shall be repeated after a delay of at least 2 etu from error detection.

When searching for a start, the receiver samples I/O periodically. The time origin being the mean between last observation of level Z and first observation of level A, the start shall be confirmed before 0,7 etu, and then ba is received at (1,5 ± 0,2) etu, bb at (2,5 ± 0,2) etu, ... bi at (9,5 ± 0,2) etu. Parity is checked on the fly.

When parity is incorrect, from (10,5 ± 0,2) etu, the receiver transmits an error signal in state A for 1 etu minimum and 2 etu maximum. The receiver then shall expect a repetition of the disputed character.

During answer to reset, this procedure, mandatory for the card, is however optional for the interface device.

Note: When searching for a start, the sampling time shall be less than 0,2 etu so that all the test zones are distinct from the transition zones.

6.1.4 Structure and Content

A reset operation brings about an answer from the card, coded over at most 32 characters : two mandatory characters TS T0, possibly followed by interface characters TAi TBi TCi TDi and "historical" characters T1 T2 ... TK.

The interface characters specify physical parameters of the integrated circuit in the card and logical particularities of the subsequent exchange protocol.

The historical characters designate for example the card manufacturer, the chip inserted in the card, the masked ROM in the chip, the state of the life of the card...

For notational simplicity, T0, TAi ..., T1 ... will designate the bytes as well as the characters in which they are contained.

See annex 1.

6.1.4.1 Structure of TS, the Initial Character

The initial character TS provides a bit sychronization sequence and defines the conventions to code data bytes in the subsequent characters. These conventions refer to ISO 1177.

The two possible values of TS (ten consecutive bits from start to bi, and corresponding hexadecimal value) are:

- A(ZZAAAAAA)Z, '3F' : ONE is A, ba is b8 (msb). Inverse conventions.

- A(ZZAZZZAA)Z, '3B' : ONE is Z, ba is b1 (lsb). Direct conventions.

With I/O initially in state Z, a bit synchronization sequence (Z)AZZA is defined which allows the interface device to determine the etu initially used by the card. An alternate definition of etu is a third of the delay between the first two falling edges in TS. Transmission and reception mechanisms in the card (including the tolerances described in sections 6.1.2 and 6.1.3) shall be consistent with this alternate definition of etu.

6.1.4.2 Structure of the Subsequent Answer to Reset

The subsequent answer to reset contains a variable number of characters in the following order : T0, the format character, TAi, TBi, TCi, TDi, the interface characters, T1,...TK, the historical characters.

The presence of the interface characters is indicated by a bit map technique as explained here after.

The presence of the historical characters is indicated by the number of bytes as specified in the format character.

TO the format character

The TO character contains two parts : the most significant half byte (b8 to b5) is named Y1 and indicates the presence of interface characters, and the least significant half byte (b4 to b1) is named K and indicates the number (0 to 15) of historical characters.

TAi TBi TCi TDi, the interface characters

Bits b5, b6, b7, b8 of the byte containing Yi (TO contains Y1 ; TDi contains Yi - 1) state whether character TAi for b5, whether character TBi for b6, whether character TCi for b7, whether character TDi for b8 are or are not (depending on whether b is 1 or 0) transmitted subsequently in this order after the character containing Yi.

When needed, the interface device shall attribute a default value to an information corresponding to a non transmitted interface character. When TDi is not transmitted, Yi-1 is null.

The interface device shall comply with these requirements in order to process commands.

T1 T2 ... TK, the historical characters

After the last interface character, when K is not null, the answer to reset is continued by K historical characters T1, T2, ... TK.

The specification of the historical characters falls outside the scope of this part of the standard.

The answer to reset is complete at the end of the guardtime of the last character.

6.1.4.3 Specifications of the Interface Characters

Among the interface bytes possibly transmitted by the card in answer to reset, this standard defines only the first 7 : TA1, TB1, TC1, TD1, TA2, TB2, TC2.

These interface bytes convey integer values either equal to or used to compute parameters (as described hereafter) the interface device shall take into account while processing commands.

Integer Values in the Interface Bytes

- TA1 codes FI, sign of M and MI over the most significant half byte (b8 to b5), the bit b4 and the three least significant bits (b3 to b1);

- TB1 codes paI, II and PI1 over the most significant (b8), the next 2 (b7 and b6) and 5 least significant (b5 to b1) bits respectively;

- TC1 codes over the eight bits N, the extra guardtime;
- TD1 codes Y2 and T over the most significant (b8 to b5) and least significant (b4 to b1) half bytes respectively;
- TA2 codes BSI over the eight bits (b8 to b1);
- TB2 codes PI2 over the eight bits (b8 to b1);
- TC2 codes WI, over the eight bits (b8 to b1);

All undefined values of the following parameters are reserved by ISO TC97/SC17 for future use.

Parameters F, D, P, pa, I, N, T, BS, W

F is the clock rate conversion factor for subsequent transmission.
D is the bit rate adjustment factor. The initial etu used during answer to reset is replaced by the work etu during subsequent transmissions in either direction.

During the answer to reset, the bit rate is 9600 bits/s on I/O when f_i provided on CLK is f_o.

During subsequent transmissions, the bit rate is D.9600 bits/s on I/O when f_s provided on CLK is $F.f_o$. The maximum values of f_s are given in table 1 below.

Initial etu : $\frac{f_o}{f_i} \cdot \frac{1}{9\,600}$ (second) Work etu : $\frac{f_o}{f_s} \cdot \frac{1}{9\,600} \cdot \frac{F}{D}$ (second)

f_i, f_s as defined in 4.2.4.

f_o is the reference frequency, characteristic of the card.

P, pa, I define the active state of VPP. Programming voltage : P volts, programming voltage accuracy : pa %, maximum programming current : I mA.

N is an extra guardtime. Before transmitting the next character, the interface device shall ensure a delay of at least (12 + N) etu from the start leading edge of the previous character.

T is a protocol type, summarizing the rules to be used to process commands.

T = 0 is the protocol described in addendum 1 ;
T = 1 is a block transmission protocol, under study as addendum 2 ;
T = 2 and T = 3 are reserved for future full duplex operations.

In a block transmission protocol (T = 1) BS indicates the maximum block size.

W is the waiting time adjustment factor. The initial waiting time used during the answer to reset is replaced by the work waiting time during subsequent commands. Initial waiting time : 9600 etu, also equal to f_o/f_i second. Work waiting time : 9600.D.W etu, equal to $F.(f_o/f_s).W$ seconds.

Default values of these parameters are:

F = D = W = 1; P = 25 volts; pa = 4 % ; I = 50 mA ; N = T = 0 ; BS = 64.

Tables

The correspondance between the parameters F, D, P, pa, I, N, T, BS, W and the integer values FI, DI, PI1, PI2, paI, II, N, T, BSI, WI is given in the following tables.

1. F, the clock rate conversion factor

FI	0	1	2	3	4	5	6	7 to 15
F	internal clock	1	1,5	2	3	4	5	RFU
f_s (max) MHz	-	4	6	8	12	16	20	RFU

2. D, the bit rate adjustment factor

$D = 2^M$

MI	0	1	2	3	4	5	6	7
M	RFU	0	1	2	3	4	5	6

Sign (b4) : value 0 means -, value 1 means -.

3. P, the programming voltage, and pa, the programming voltage accuracy

PI1 from 5 to 25 gives the value of P in volts. PI1 = 0 indicates that VPP is not connected in the card which generates an internal programming voltage from VCC. Other values of PI1 are undefined.

When PI2 is present, the indication of PI1 should be ignored. PI2 from 50 to 250 gives the value of P in 0,1 volt. Other values of PI2 are undefined.

paI = 0 indicates pa = 4 % ; paI = 1 indicates pa = 2,5 %.

4. I, the maximum programming current

II	0	1	2	3
I mA	25	50	100	200

5. N, the extra guardtime

N codes directly the extra guardtime. from 0 to 255 in etu.

6. T, the protocol type

T codes the protocol type, with the values 0 to 15.

7. BS, the block size

BSI = 0 indicates 256 bytes ; BS is equal to BSI from 1 to 255.

8. W, the waiting time adjustment factor

W = WI/10 (W in the range from 0,1 to 25,5 seconds).

6.2 Answer to Reset in a Synchronous Transmission

6.2.1 Clock frequency and bit rate

There is a linear relationship between the bit rate on the I/O line and the clock frequency provided by the interface device on CLK.

Any clock frequency between 7 KHz and 50 KHz may be chosen for the reset sequence. A clock frequency of 7 KHz corresponds to 7 Kbit/s, and values of the clock frequency up to 50 KHz cause corresponding bit rates to be transmitted.

6.2.2 Structure of the header of the answer to reset

The reset operation results in an answer from the card containing a header transmitted from the card to the interface.

The header has a fixed length of 32 bits and begins with 2 mandatory fields of 8 bits, H1 and H2.

The chronological order of transmission of the information bits shall correspond to bit identification b1 to b32 with least significant bit transmitted first. The numerical meaning corresponding to each information bit considered in isolation is that of the digit:

0 for a unit corresponding to state A (space)
1 for a unit corresponding to state Z (mark)

6.2.3 Timing of the header

After the reset procedure, see section 5.2 and figure 2, the output information will be controlled by clock pulses. The first clock pulse is applied between 10 us and 100 us (t_{14}) after the falling edge on RST to read the data bits from the card. The clock pulses have a high state whose time can be varied between 10 us and 50 us (t_{15}) and a low state with a time between 10 us and 100 us (t_{16}).

The first data bit is obtained on I/O while clock is low and is valid 10 us (t_{13}) at least after the falling edge of RST. The following data bits are valid 10 us (t_{17}) at least after the falling edge of CLK. Each data bit is valid until the next falling edge of the following clock pulse on CLK. The data bits can therefore be sampled at the rising edge of the following clock pulses.

6.2.4 Data content of the header

The header first allows a quick determination of whether the card and the interface device are compatible. Secondly it enables cards with small memory capacities to use a protocol according to this standard.

If there is no compatibility between the header and the interface device, the contacts shall be deactivated according to section 5.4.

6.2.4.1 Data content of field H1

The first field H1 codes the protocol type. The values of the codes and the corresponding protocol types are:

Hexadecimal value	Protocol type
00 and FF	not to be used
01 to FE	each value is assigned by ISO TC97/SC17 to one protocol type

6.2.4.2 Data content of field H2

The second field codes parameters for the protocol type coded in field H1. The values of H2 are to be assigned by ISO TC97/SC17.

6.2.4.3 Data content of the remaining fields

The specifications of the remaining fields fall outside the scope of this part of the standard. The role of these remaining fields is similar to that of the historical characters mentioned in 6.1.4.

ANNEX 1

DIAGRAMS RELATED TO ANSWER TO RESET

(This annex does not form part of the standard)

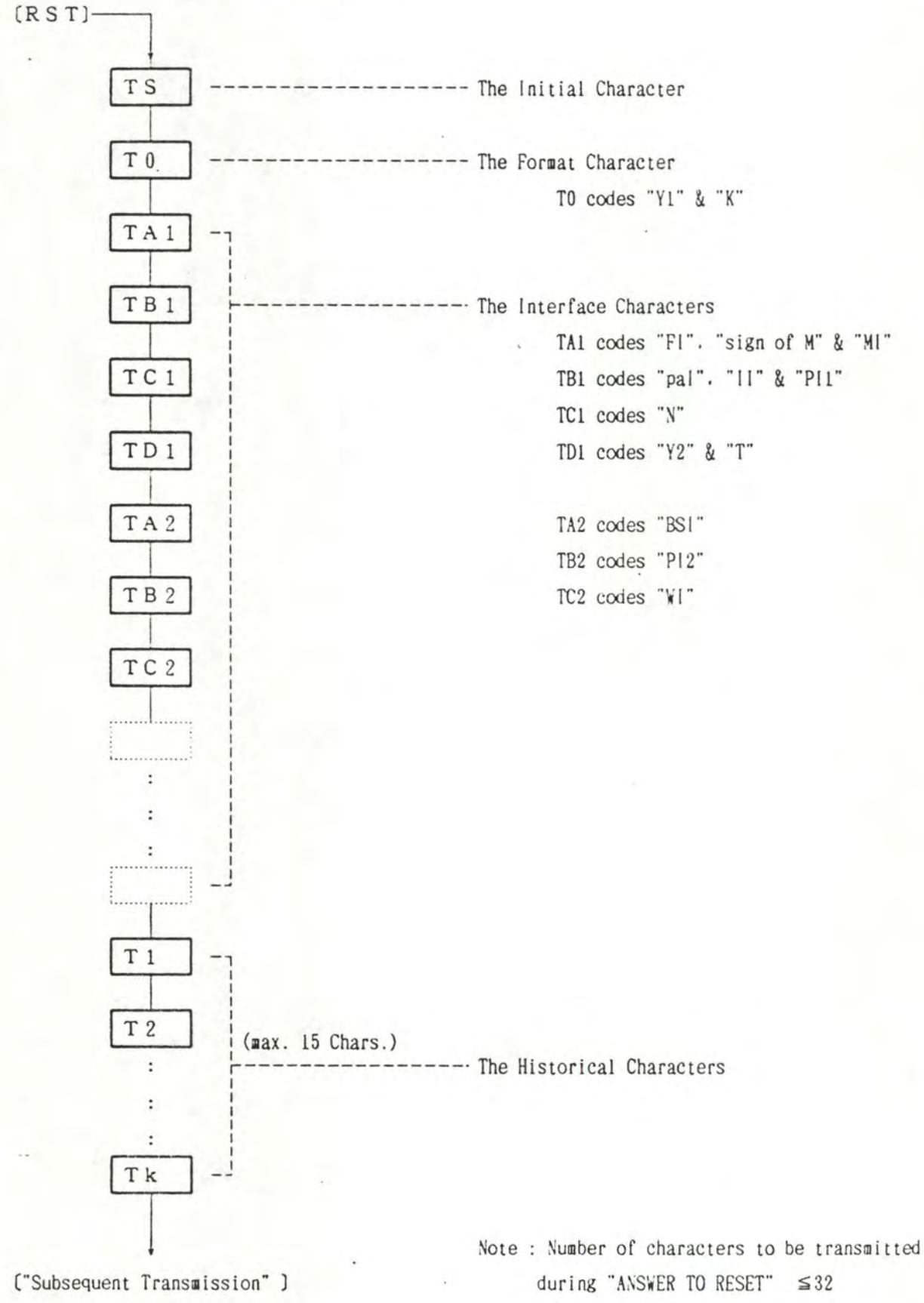
Introductory note :

This annex includes the diagrams on the specification of "ANSWER TO RESET" in an Asynchronous Transmission.

The Protocol Type "T" defined in TD1 and all undefined values of the Parameters are reserved by ISO/TC 97/SC 17 for future use.

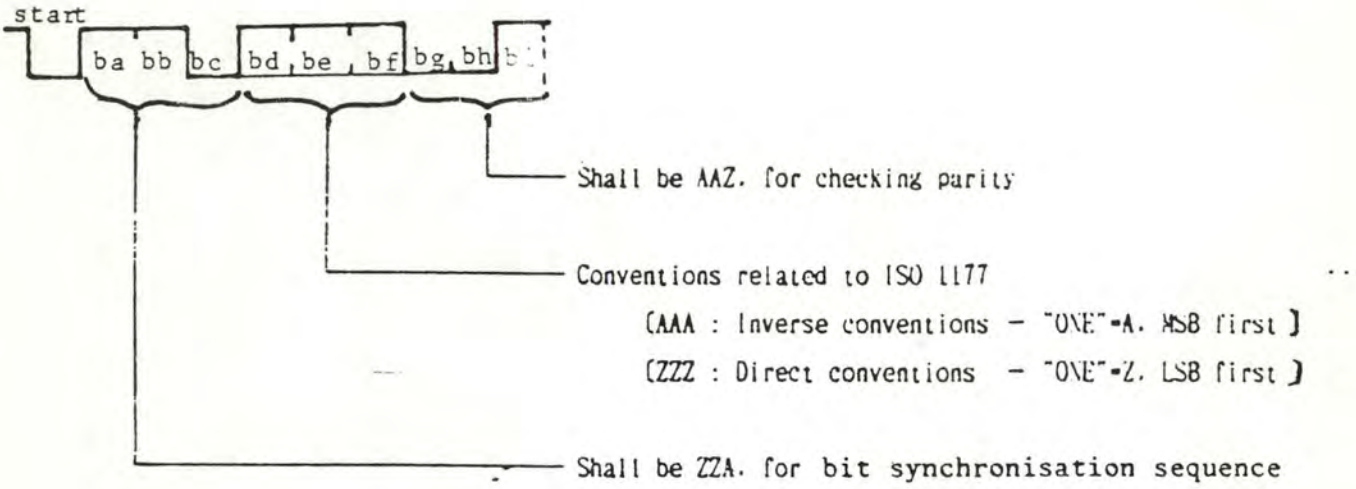
Contents :

I. The Configuration of "ANSWER TO RESET"



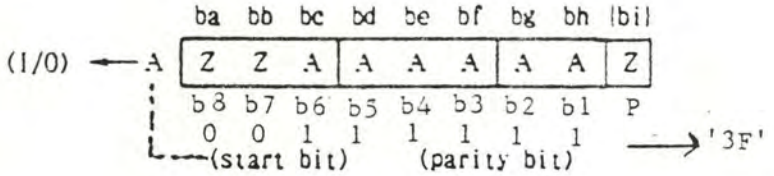
Note : Number of characters to be transmitted during "ANSWER TO RESET" ≦32

II. The Information provided by the Initial Character : TS

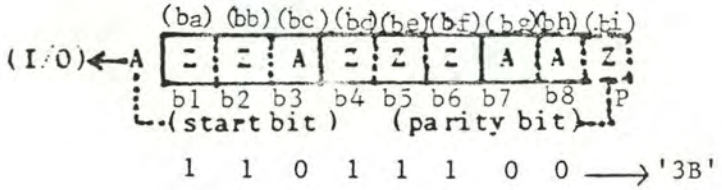


- Following two values are defined. as an internal expression in the card. for reproducing the above information in the interface device.

— Internal value = '3F' : for the cards based on MSB first logic (where "ONE" is A)

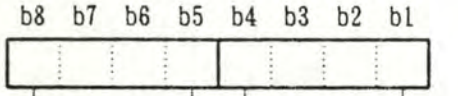


— Internal value = '3B' : for the cards based on LSB first logic (where "ONE" is Z)



Note : The internal expression in the card is assumed for all following diagrams. The interface device can receive these characters, according to the conventions specified by "TS" (the Initial Character).

Ⅲ. The Information provided by the Format Character : TO (for "Y1" and "K")



"K" : Number of the Historical Characters
〔K = 0～15〕

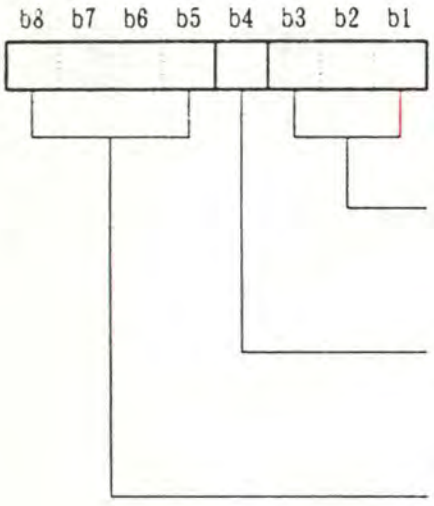
"Y1" : the indicator for the presence of the Interface Characters
〔TA1 is transmitted, when b5=1 〕
〔TB1 is transmitted, when b6=1 〕
〔TC1 is transmitted, when b7=1 〕
〔TD1 is transmitted, when b8=1 〕

Note-1 : The default values are used in the interface device, for the parameters specified in the Interface Characters not transmitted.

Note-2 : b8 shall be 1 for the definition of the Protocol Type "T", execpt when "T" = 0 is selected.

Ⅳ. The Information provided by the Interface Charcters : TAi—TDi

Ⅳ-1 TA1 (The Interface Character for "FI", "sign of M" and "MI")



"MI" : the integer value for the index "M", to obtain the bit rate adjustment factor "D" [default: MI = 1]

"sign of M" : the indicator for the sign of the index "M" [default: sign of M = 0]

"FI" : the integer value for the clock rate conversion factor "F" and for the maximum frequency f_s (max) [default: FI = 1]

[Value of "F" and f_s (max) obtained from "FI"]

"FI"	0	1	2	3	4	5	6	7 to 15
"F"	internal clock	1	1.5	2	3	4	5	RFU
f_s (max) MHz	-	4	6	8	12	16	20	RFU

[Value of "M" obtained from "MI" and "sign of M"] ⟶ $D = (2^M)$

	"sign of M" = 1								"sign of M" = 0						
"MI"	7	6	5	4	3	2	1	0	1	2	3	4	5	6	7
"M"	-6	-5	-4	-3	-2	-1	0	RFU	0	1	2	3	4	5	6

Note-1 : f_s is the frequency provided by the interface device in "Subsequent Transmission", and its maximum value is given by f_s (max) in the above table.

Note-2 : The bit rate adjustment factor "D" is obtained by the formula $D = (2^M)$ and the above table.

Note-3 : Initial etu in second (etu in "ANSWER TO RESET")

$$= (f_0 / f_i) \cdot (1/9600)$$

where f_0 is the reference frequency required by the card to generate 9600 bits/s. and the value is 3.579545 MHz

f_i is the initial frequency provided by the interface device during "ANSWER TO RESET".

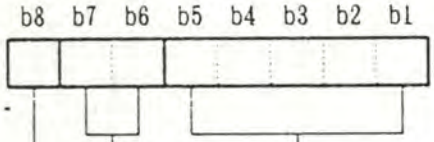
Note-4 : Work etu in second (etu in "Subsequent Transmission")

$$= (f_0 / f_s) \cdot (1/9600) \cdot (F / D)$$

where f_s is the frequency provided by the interface device during "Subsequent Transmission".

Work etu = (Initial etu / D). when $f_s = F \cdot f_i$

Ⅳ-2 TB1 (The Interface Character for "pal", "II" and "PII")



"PII" : the integer value for the active state of the programming voltage, "P" [default: PII = 25]
[P = PII (Volt). where PII = 0 or 5~25]

"II" : the integer value for the maximum current, "I", at the programming voltage [default: II = 1]
[Value of "I" obtained from "II"]

"II"	0	1	2	3
"I" (mA)	25	50	100	200

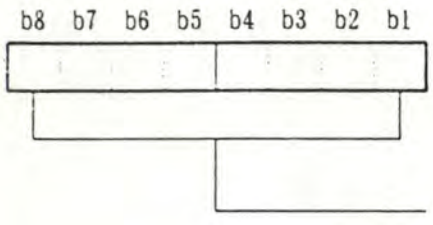
"pal" : the integer value for the programming voltage accuracy
[pa = ±4%. when pal = 0] [default: pal = 0]
[pa = ±2.5%. when pal = 1]

Note-1 : V_{PP} is generated from V_{CC} internally. when "PII" = 0.

Note-2 : The value of "I" can be obtained by the following formula (for reference).

$I = 25 \cdot (2^{II})$ (mA). where II = 0, 1, 2 or 3

Ⅳ-3 TC1 (The Interface Character for "N")

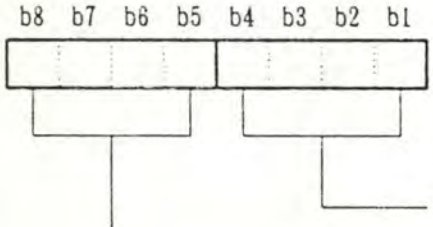


"N" : the extra guardtime assigned by the card to the interface device [default: N = 0]

(minimum delay=12+N (etu) where N = 0~255)

Note : No extra guardtime is assigned by the interface device to the card.

Ⅳ-4 TD1 (The Interface Character for "Y2" and "T")



"T" : the Protocol Type for "Subsequent Transmission" [default: T = 0]

(T = 0 : half-duplex character transmission)

(T = 1 : half-duplex block transmission)

(T = 2 & 3 : full-duplex transmission)

"Y2" : the indicator for the presence of the Interface Characters

(TA2 is transmitted, when b5 = 1)

(TB2 is transmitted, when b6 = 1)

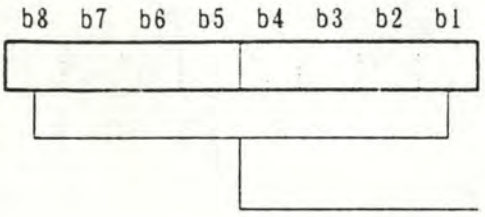
(TC2 is transmitted, when b7 = 1)

(TD2 is transmitted, when b8 = 1)

Note-1 : All other values of "T" are reserved by ISO/TC 97/SC 17 for future use.

Note-2 : "Y2" = 0000 (binary) is assumed, when TD1 is not transmitted.

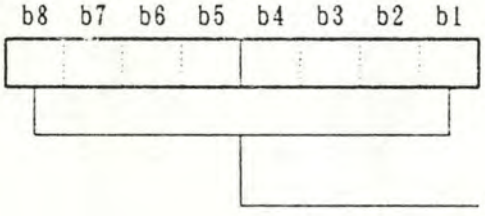
Ⅳ－5 TA2 (The Interface Character for "BSI")



"BSI" : the integer value for the block size, "BS", in bytes
[default: BSI = 64]
[BS = BSI, except when BSI=0]
[BS = 256, when BSI=0]

Note : The value of "BS" indicates the maximum number of characters included in one block.

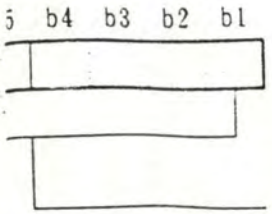
Ⅳ－6 TB2 (The Interface Character for "PI2")



"PI2" : the integer value for the active state of the programming voltage, "P"
[P = 0.1 • PI2 (Volt), where PI2 = 50～250]

Note : The old value of "P", defined by "PI1" in TB1, should be replaced with this value, when "PI2" is present. (The default value is not specified for this parameter.)

(The Interface Character for "Wl")



"Wl" : the integer value for the waiting time adjustment factor "W" . [default: Wl = 10]

[waiting time = F · (f_0 / f_s) · W (second)]

[where W = Wl / 10]

ıe delay from the start leading edge of a character to the start leading
the following character shall be less than the waiting time:
ı at F.fo .

Characters : T1−Tk

specification on these additional characters falls outside the scope of this
of the standard.

DRAFT PROPOSAL ISO/DP 7816/3 ADD.1	
date 1986 11 10	ISO/TC 97/ SC 17
superseded document 97/17/4 N 245	

THIS DOCUMENT IS STILL UNDER STUDY AND SUBJECT TO CHANGE. SHOULD NOT BE USED FOR REFERENCE PURPOSES.

ISO/TC 97 SC 17
" IDENTIFICATION CARDS "

Secretariat B.S.I.

Circulated to P- and O-members, technical committees and internatio organizations in liaison for :
- discussion at
- comments by
- voting by (P-members only)

Title IDENTIFICATION CARDS - INTEGRATED CIRCUIT(S) CARD WITH CONTACTS
PART 3: ELECTRONIC SIGNALS AND EXCHANGE PROTOCOLS
ADDENDUM 1: STRUCTURE AND PROCESSING OF COMMANDS IN AN ASYNCHRONOUS TRANSMISSION

Introductory note

Addendum 1 - Structure and Processing of Commands in an Asynchronous Transmission

Always initiated by the interface device, a command tells the card what to do in a 5-byte header, and allows a transfer of data bytes under control of procedure bytes sent by the card.

It is assumed that the card and the interface device know a-priori the direction of data, in order to distinguish between instructions for incoming data transfers (where data enter the card during execution) and instructions for outgoing data transfers (where data leave the card during execution). During the process of a command, the procedure of error detection and character repetition described in 6.1.3 is mandatory both in the card and in the interface device.

Interindustry commands for interchange will be defined in a subsequent part of this standard.

1. The command header sent by the interface device

The interface device transmits a header over five successive bytes designated CLA, INS, A1, A2, L.

- CLA is an instruction class.
- INS is an instruction code in the instruction class. The instruction code is valid only if the least significant bit is 0, and the most significant half-byte is neither '6' nor '9'.
- A1, A2 are a reference (eg. an address) completing the instruction code.
- L is the number of data bytes (D1...DL) which are to be transmitted during the command. The direction of movement of these data is a function of the instruction. In an outgoing data transfer command, L=0 introduces a 256-byte data transfer from the card.

After transmission of such a 5-byte header, the interface device waits for a procedure byte.

2. The procedure bytes sent by the card

The values of the procedure bytes request action by the interface device. Three types of procedure bytes are specified:

- ACK (the seven most significant bits in an ACK byte are all equal or all complementary to those in the INS byte, apart from the values '6X' and '9X'). The interface device controls VPP and exchanges data depending on ACK values.
- NUL (='60'). The interface device resets its timer and waits for a new procedure byte, without taking any further action on VPP or on data.
- SW1 (='6X' or '9X', except '60'). The interface device maintains or sets VPP at idle and waits for a SW2 byte to complete the command.

Any transition of VPP (active/idle) must occur within guardtime of the procedure byte, or on timer overflow. At each procedure byte, the card can proceed with the command by an ACK or NUL byte, or show its disapproval by becoming unresponsive, or conclude by an end sequence SW1-SW2.

2.1 Acknowledge bytes

The ACK bytes are used to control VPP state and data transfer.

- When exclusive-ORing the ACK byte with the INS byte gives '00' or 'FF', the interface device maintains or sets VPP at idle.
- When exclusive-ORing the ACK byte with the INS byte gives '01' or 'FE', the interface device maintains or sets VPP at active.

- When the seven most significant bits in the ACK byte have the same value as those in the INS byte, all remaining data bytes, DI...DL, if any remain, are transferred subsequently.
- When the seven most significant bits in the ACK byte are complementary to those in the INS byte, only the next data byte, DI, if one remains, is transferred.

After these actions, the interface device waits for a new procedure byte.

2.2 NULL byte (='60')

After resetting its timer, the interface device waits for a new procedure byte.

2.3 Status bytes (SW1 = '6X' or '9X', except '60'; SW2 = any value)

The end sequence SW1-SW2 gives the card status at the end of the command.

The normal ending is indicated by SW1-SW2 = '9000'.

When the most significant half-byte of SW1 is '6', the meaning of the SW1 is independant of the application. Five of these have a special meaning:

'6E' : The card does not support the instruction class
'6D' : The instruction code is not programmed or is invalid
'6B' : The reference is incorrect
'67' : The length is incorrect
'6F' : No precise diagnosis is given

Other values are reserved for future use by TC97/SC17

When the SW1 is neither '6E' nor '6D', the card supports the instruction.

This part of the standard interprets neither '9X' SW1 bytes, nor SW2 bytes; their meaning relates to the application itself.

ANNEX 1

THIS ANNEX DOES NOT FORM PART OF THE STANDARD

Processing of a Command in a Asynchronous Transmission

The command which specifies the function to be executed in the card is issued and transmitted by the interface device. By a procedure sequence the card controls the data transfer. The respond which specifies the result of command execution is issued and transmitted from the card.

1. Structure of Sequence

1-1 Basic Structure

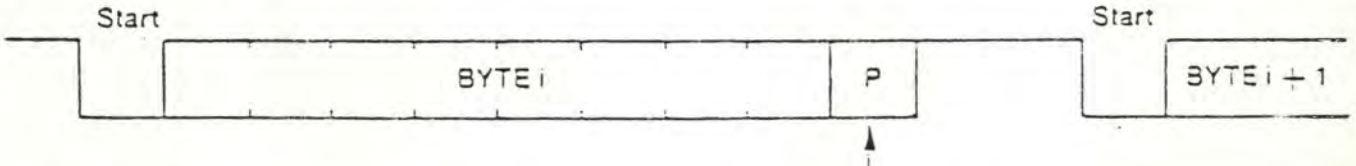
A sequence in the asynchronous transmission consists of several fields such as :

CLA : Instruction class.
INS : Instruction code.
REF : Reference, consists of two bytes A1, A2.
L : Number of bytes to be transmitted in the Data Field.
PB : Procedure byte.
DATA : Data.
STB : Status byte.

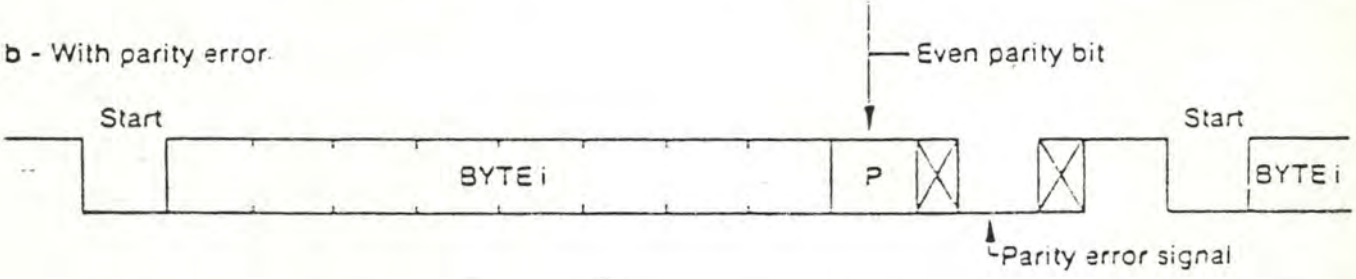
Each field consists of a single byte or several bytes. On the transmission line, a byte in a field is transmitted in the character describe below.

Figure 1 - Byte transmission diagram

a - Without parity error



b - With parity error



Before transmitting the next character, the interface device shall ensure a delay of at least (12 + N) etu from the start leading edge of the previous character.

N is given by the interface byte TC1.

1-2 The Command Block

5/6

Interface Device → Card				
		REF		
CLA	INS	A1	A2	L

1-3 Procedure Sequence

1-3-1 Ack Command Block

C → IO
PS
ACK

C means Card

IO means Interface Device

1-3-2 Request waiting for a new procedure sequence

C → IO
PS
NULL

Note : The different values of the procedure byte are specified in section 2.

1-4 Data Sequence

A data transfer is subsequent to an Ack Command Block.

1-4-1 Data Character

IO = C (1)
DATA
Di

1-4-2 Data Block

Interface Device = Card (1)			
Data			
Di	Di + 1		DL

(1) The direction of the transfer depends on the Function to be executed.

1-5 End of Command Block

C ⟶ ID	
PB	STB
SW1	SW2

2. Procedure byte and Status code

2-1 Procedure byte

The values of the procedure bytes request action by the interface device

	Value	Action
ACK	INS	Vpp is idle. All remaining data bytes are transferred subsequently.
	INS + 1	Vpp is activate. All remaining data bytes are transferred subsequently.
	$\overline{INS}$	Vpp is idle. Next data byte is transferred subsequently.
	$\overline{INS + 1}$	Vpp is activate. Next data byte is transferred subsequently.
NULL	60	The interface device waits for a new procedure byte. No further action on Vpp
SW1	SW1	Vpp is idle - the interface device waits for a SW2 byte.

2-2 Status Code

The End of Command Block gives the card status at the end of the command.

When SW1 = 6X, the meaning of SW1 is independant of the application.

Code	Meaning
6E	The card does not support the instruction class
6D	The instruction code is not programmed or is invalid.
6B	The reference is incorrect.
67	The length is incorrect.
6F	No precise diagnosis is given.

The standard interprets neither '9X' SW1 bytes, or SW2 bytes. Their meaning relates to the application itself.

The normal ending is indicated by SW1-SW2 = '9000'

ANNEXE 2

TYPES DE DONNEES SPECIFIQUES AU LOGICIEL

A2.1. BCD code

Binary Coded Decimal (BCD) interprets 4 at a time as a decimal number. The bit pattern and its corresponding decimal character is listed below.

Bit pattern	BCD character
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9

A2.2. ISO 5 code

International Standards Organizations 5 bits (ISO 5) character code interprets 5 at a time as a character. The bit pattern and its corresponding character is listed below.

Bit pattern	ISO 5 character	Bit pattern	ISO 5 character
00000	@	10000	P
00001	A	10001	Q
00010	B	10010	R
00011	C	10011	S
00100	D	10100	T
00101	E	10101	U
00110	F	10110	V
00111	G	10111	W
01000	H	11000	X
01001	I	11001	Y
01010	J	11010	Z
01011	K	11011	[
01100	L	11100	\
01101	M	11101	]
01110	N	11110	^
01111	O	11111	_

A2.3. ISO 6 code

International Standards Organizations 6 bits (ISO 6) character code interprets 6 at a time as a character. The bit pattern and its corresponding character is listed below.

Bit pattern	ISO 6 character	Bit pattern	ISO 6 character
000000	space	100000	@
000001	!	100001	A
000010	"	100010	B
000011	#	100011	C
000100	$	100100	D
000101	%	100101	E
000110	&	100110	F
000111	'	100111	G
001000	(	101000	H
001001	)	101001	I
001010	*	101010	J
001011	+	101011	K
001100	,	101100	L
001101	-	101101	M
001110	.	101110	N
001111	/	101111	O
010000	0	110000	P
010001	1	110001	Q
010010	2	110010	R
010011	3	110011	S
010100	4	110100	T
010101	5	110101	U
010110	6	110110	V
010111	7	110111	W
011000	8	111000	X
011001	9	111001	Y
011010	:	111010	Z
011011	;	111011	[
011100	<	111100	\
011101	=	111101	]
011110	>	111110	^
011111	?	111111	_

A2.4. ASCII code

The American Standard Code for Information Interchange(ASCII) character code interprets 7 at a time as a character. The bit pattern and its corresponding character is listed below.

Bit pattern	ASCII character	Bit pattern	ASCII character
0000000	null	1000000	@
0000001		1000001	A
0000010		1000010	B
0000011		1000011	C
0000100		1000100	D
0000101		1000101	E
0000110		1000110	F
0000111	beep	1000111	G
0001000		1001000	H
0001001	tab	1001001	I
0001010	line feed	1001010	J
0001011	home	1001011	K
0001100	form feed	1001100	L
0001101	CR	1001101	M
0001110		1001110	N
0001111		1001111	O
0010000		1010000	P
0010001		1010001	Q
0010010		1010010	R
0010011	!!	1010011	S
0010100		1010100	T
0010101		1010101	U
0010110		1010110	V
0010111		1010111	W
0011000		1011000	X
0011001		1011001	Y
0011010		1011010	Z
0011011		1011011	[
0011100	cursor right	1011100	\
0011101	cursor left	1011101	]
0011110	cursor up	1011110	^
0011111	cursor down	1011111	_
0100000	space	1100000	`
0100001	!	1100001	a
0100010	"	1100010	b
0100011	#	1100011	c
0100100	$	1100100	d
0100101	%	1100101	e
0100110	&	1100110	f
0100111	'	1100111	g
0101000	(	1101000	h
0101001	)	1101001	i
0101010	*	1101010	j
0101011	+	1101011	k
0101100	,	1101100	l
0101101	-	1101101	m
0101110	.	1101110	n
0101111	/	1101111	o

0110000	0	1110000	p
0110001	1	1110001	q
0110010	2	1110010	r
0110011	3	1110011	s
0110100	4	1110100	t
0110101	5	1110101	u
0110110	6	1110110	v
0110111	7	1110111	w
0111000	8	1111000	x
0111001	9	1111001	y
0111010	:	1111010	z
0111011	;	1111011	{
0111100	<	1111100	|
0111101	=	1111101	}
0111110	>	1111110	~
0111111	?	1111111	